COMBAT IDENTIFICATION MODELING USING ROBUST OPTIMIZATION TECHNIQUES

THESIS

TaeHo Kim, Captain, ROKA

AFIT/GOR/ENS/08-11

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GOR/ENS/08-11

COMBAT IDENTIFICATION MODELING USING ROBUST OPTIMIZATION TECHNIQUES

THESIS

Presented to the Faculty

Department of Operational Sciences

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Operations Research

TaeHo Kim, BS

Captain, ROKA

March 2008

AFIT/GOR/ENS/08-11

COMBAT IDENTIFICATION USING ROBUST OPTIMIZATION TECHNIQUES

TaeHo Kim, BS
Captain, ROKA

Approved:

Dr. Kenneth W. Bauer (Chairman)	Date
Dr. J. O. Miller (Member)	Date

AFIT/GOR/ENS/08-11

Abstract

The purposes of this research were: (1) the modeling of a CID situation and (2) the search for robust and controllable input variable settings. The inputs were defined as controllable and noise variables and the confusion matrices in ROC theory were adapted to act as controllable factors. In this research a simple virtual battlespace representation is employed. The experimental results of the CID system are summarized by a posterior confusion matrix and throughout the confusion matrix analysis we can obtain all various types of data such as accuracy, error cost, error rates, and so forth. To find the optimal parameters three evaluation techniques were applied: (1) Linearly constrained discrete optimization, (2) Taguchi's S|N ratio method and (3) Robust parameter design with a combined array. The results are compared and contrasted across different objective functions.

In conclusion, if we consider the diverse characteristics of CID, the simulator needs to focus on finding the controllable parameter that yields the maximum accuracy value. This is because the minimum cost is typically accomplished at the point of maximum accuracy and the cost approach is very subjective depending on the decision maker and battlefield situation. In addition, the most preferable evaluation method is RPD with a combined array due to its superior performance outside of the design space. In the final analysis, we need a detector/classifier that has good performance to minimize error costs and maximize label accuracy.

AFIT/GOR/ENS/08-11

Acknowledgments

I would like to thank Dr. Kenneth W. Bauer for his direction and guidance in this thesis effort. His knowledge of Combat Identification and analysis techniques were instrumental in my ability to quantify my findings in a new manner. Also, I am grateful to him for enduring the countless revisions to my document. I would also like to thank Dr. J.O. Miller in his seemingly endless knowledge and resources in Combat simulation. Finally, I thank to my wife, my healthy baby and all my American friends. Their existence and support gave me the motivation and improvement to proceed.

Table of Contents

Page

Page

List of Figures

Page

List of Tables

COMBAT IDENTIFICATION MODELING USING
ROBUST OPTIMIZATION TECHNIQUES

I. Introduction

Background

There is a phrase in The Art of War, written by Sun Tzu in 600BC: *"So it is said that if you know your enemies and know yourself, you will fight without danger in battles. If you only know yourself, but not your opponent, you may win or may lose. If you know neither yourself nor your enemy, you will always endanger yourself."* In Sun Tzu's view, the goal of war is not only to win but also to win without friendly casualties. What do we need to accomplish his thought?

There is another expression in the book of Morals and Conduct, written by Ibn Hazm of Codorva (994-1064): "*The measure of prudence and resolution is to know a friend from an enemy; the height of stupidity and weakness is not to know an enemy from a friend"*. In addition to Sun Tzu's suggestion, Ibn Hazm emphasizes an identification friend from enemy and enemy from friend. The kind of information on an enemy can be very diverse but the most important information that we have to get is knowledge of the position of both the enemy and the friendly force. We usually use radar and communication skills to confirm the location of a friendly force and to classify characteristics between enemy and friendly forces. However, in thinking about the real battlefield situation, we cannot be sure whether the required information is always available or not.

On 10 August 2006, in an interview in Secretary Rumsfeld's office, the Secretary said, "*I was asked that when I was up at the confirmation hearings in January of '01, and I said intelligence. And if you think about this department, we have just enormous capability to finish. If you use the phrase "find, fix and finish," we can finish something if we can find it and fix it in time and location. The problem is finding it.*" [1]. Obviously, there is a difference between identification and location but if we think about the procedure of location, it has to include the entire identification process.

Successful Combat Identification (CID) is an essential factor to achieve various objectives in combat. For example, a detection of an enemy target to destroy it and the classification of a friendly force to avoid fratricide in the complex battlefield can be achieved by a successful CID process. A good method to increase confidence in CID is iterative CID processing. This is based on accumulated information about the ROI. One of the ways to assess iterative running of the new CID process is simulation. According to Law and Kelton, simulation is a "useful and powerful tool" to aid in "evaluating military systems."

Research Problem

In the fall of 1994, a DoD Combat CID Study was formed at the request of Dr. Paul Kaminski to do a Department wide review of CID. And this study was completed by the summer of 1995 [2]. The Defense Science Board Task Force concluded that there was no crisis in CID calling for extraordinary action and suggested the maintaining of current CID budgets and activities [2:45-47]. After the Task Force's report, CID has been investigated considerably, especially with respect to ATR. The study of the ATR model has been conducted by Dr. Bauer and his students at AFIT. But we desire a full process model of CID including ATR. Therefore the model in this research may be among the first attempts at CID simulation. Due to this first trial of CID simulation, there is an absence in the literature in regard to CID simulation. Thus it is necessary to start this research from the study of ATR and we expect to see connections between the previous ATR system and this research.

Research Objective

In this paper we first need to construct a general, basic and reliable CID model such a model does not exist. To make the CID model, we use several assumptions such as predetermined ROC curves, the number of enemies and so forth. In addition, we try to get the robust parameter settings that minimize cost and maximize label accuracy by employing several techniques: (1) Linearly constrained discrete optimization (LCDO), (2) Taguchi's S|N ratio method with a crossed array design, and (3) Robust parameter design (RPD) with a combined array. After modeling and evaluation, we will examine our results through confirmation experiments and consider their strengths and weaknesses.

Scope

In exploring the question of CID simulation, this paper will mainly deal with fundamental CID modeling and its evaluation technique. It is not within the scope of this study to outline CID in its entirety and it may be impossible since CID itself has many uncertainties and depends on the battlefield situation. Thus, in this research we focus on constructing a CID simulation and motivating further research.

Overview

The next four Chapters provide detailed information and descriptions of this research. Chapter II summarizes the literature relating directly to this research. Chapter III explains the CID model established for this research and outlines the methodology used to perform the problem discussed in Section 1 and Section 2. Chapter IV presents the description of experiments and the results of the analysis. Chapter V provides the author's conclusions and recommendations for future research.

II. Literature Review

Overview of Department of Defense Modeling and Simulation Pyramid

Modeling and Simulation (M&S) is defined as "The process of designing a model of a system and conducting experiments with this model for the purpose either of understanding the behavior of the system or of evaluating various strategies for the operation of the system" [3]. There are many reasons why a system is modeled by computer simulation. For example, a specific system may be too expensive or dangerous to use in the real world, the system may not exist, or the system may have inherent variability that causes very messy or completely intractable analysis of its processes. Especially, it would be unethical and inhuman to start real combat merely to test a new weapon system.

A model of a real system is a representation of some of the components of the system and of some of their actions and interrelationships that are useful for describing or predicting the behavior of the system [5: Sec I, 1]. The goal in the design of combat models is to represent warfare as appropriately and accurately as possible, because results from combat simulation can lead to national defense determinations.

Model Hierarchy

Within the field of DoD M&S, Combat models are classified into a multi-tiered or hierarchical family of models. This categorization is described as a pyramid, with higher resolution combat models involved with a small-scale of battle at the lowest position and large-scale of combat models at the highest position.

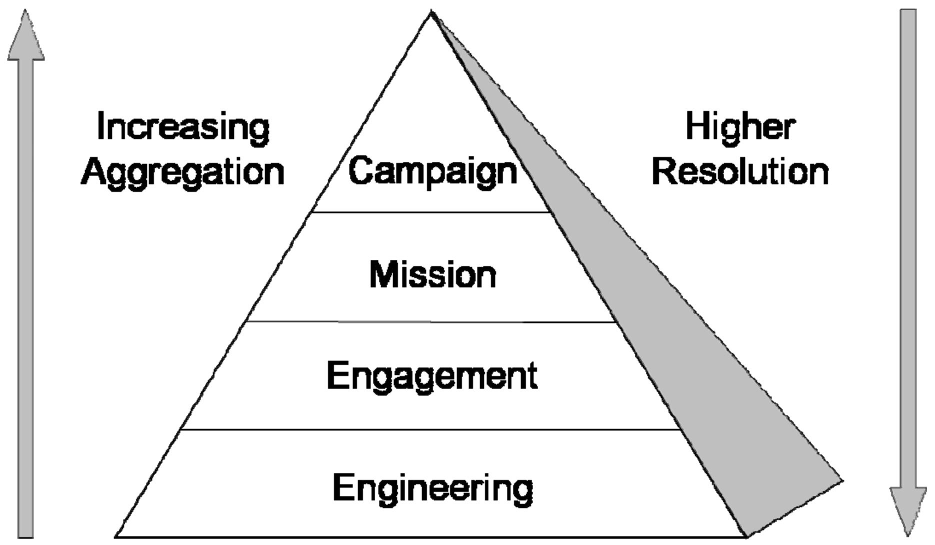


Figure 1: DoD M&S Pyramid [20]

At the bottom are the Engineering models, which describe or determine individual characteristics of systems. An example of this would be the accuracy of the given sensor with a particular condition of weather. At the second level of the pyramid are the single Engagement level models, which determine performance of systems against adversary systems. These models are used at the operational level with rule sets controlling the behavior of systems within simulation. An instance of this would be a missile fly-out model. Next on the way up the pyramid are the Mission level models, which move from single-unit engagement to multiple-unit engagement, e.g., a F-16 flying a Suppression of Enemy Air Defenses (SEAD) mission, or a squadron of F-16s flying SEAD against an Integrated Air Defense System (IADS). At the top of the pyramid are the campaign level models, which involve a lot of different systems. They are simulated with low resolution and high aggregation due to many entities and their interactions. In general, the aggregation decreases as one moves down the pyramid, and resolution increases as one moves down, but sometimes this is not true. By aggregation it is meant that each entity is not simulated but is replaced by probabilities and percentages. The primary model used in this research effort, CID, is primarily an engagement and mission level model, and will be described in detail later in this literature review.

Description of CID Mission

Definition

CID is the process of attaining an accurate characterization of entities in a combatant's area of responsibility to the extent that high-confidence, real-time application of tactical options and weapon resources can occur. The objective of CID is to maximize combat/mission effectiveness while reducing total casualties due to enemy action and fratricide [5:1].

Importance of Effective CID

According to the *Report of the Defense Science Board Task Force on CID* (May 1996), effective CID is critical for achieving improved operational effectiveness and reducing fratricide. Fratricide may not determine the result of conflicts in the foreseeable future; however, casualties due to fratricide are considered much less acceptable than those caused by enemy attack. Furthermore, fratricide has crucial side effects such as discouraging morale, potential loss of confidence due to a fear of fratricide and loss of public support. Figure 2 shows how correct or incorrect CID can affect the outcome of a conflict. If an object is hostile, but not ID'ed as hostile, and therefore the Blue force does not destroy it, the Blue force's ships and crews may be lost and perhaps worse, eventually wars are lost. Further, if the object is friendly or neutral and the Blue force destroys it due to a false ID, then lives are lost and wars may be started.

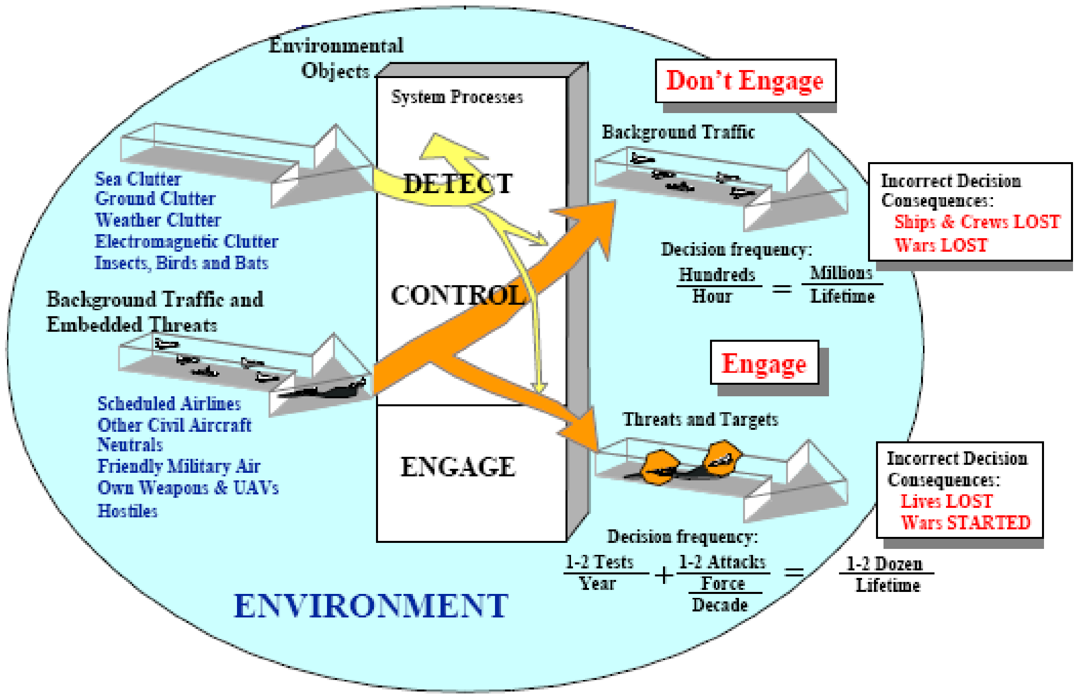


Figure 2: Importance of Effective CID [6:4]

CID is, in fact, a complex requirement for the armed services to accomplish the objectives of each case. In order to maximize the efficiency of weapons on the asymmetric battlefield while protecting friendly forces, warfighters must use a combination of on-board Cooperative and Non-cooperative Identification systems, along with Tactics, Techniques, and Procedures (TTP) that maximize systems' capabilities.

Figure 3 shows why CID processes have to be important in the battlefield. The CID process is located in the middle of the engagement process; it means consistent development in CID is essential to progress the engagement process. Figure 3 also explains that CID is beneficial in combat for even more than preventing fratricide. In effect, early development of CID was for battle management purposes so that the military commander could control and intentionally plan and execute the battle with good knowledge of his forces [2:4].

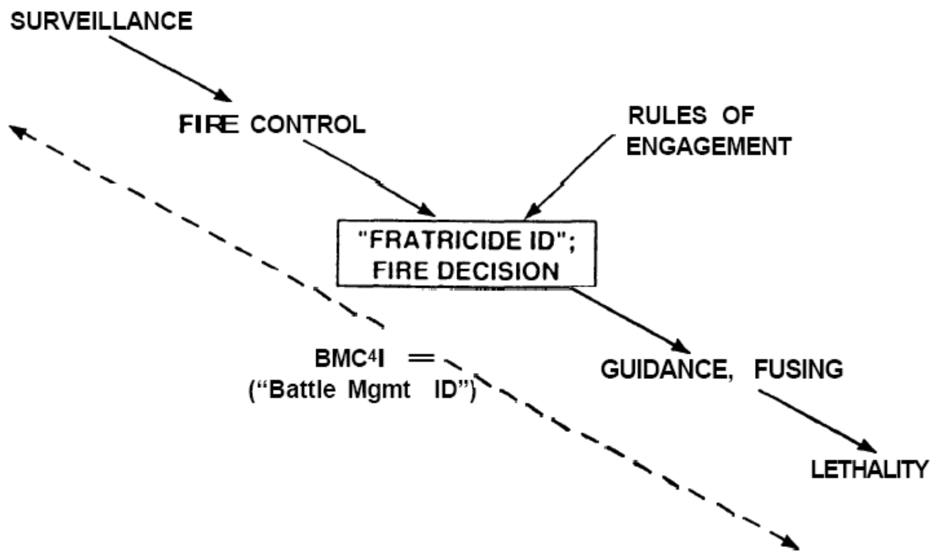


Figure 3: The Target Engagement Process [2:4]

Outline of the CID Process

The major stages of the CID process are: Detection, Cooperative Identification, Non-cooperative Identification (ATR), Fusion, Decision and Firing. If one envisions the battle field as a grid over which blue/red/neutral targets and clutters have been placed, then the Blue force performs a sequential CID cycle as follows:

- Examines each grid element for detections
- Assigns Cooperative labels to indicated grid points

- Turns on the ATR system and makes Non-cooperative assignments to all detections and fuses everything (There may be a secondary sensor somewhere doing similar things)
- Decides whether to shoot at a detected and classified target based on the indication of the two sensors (see figure 4.)

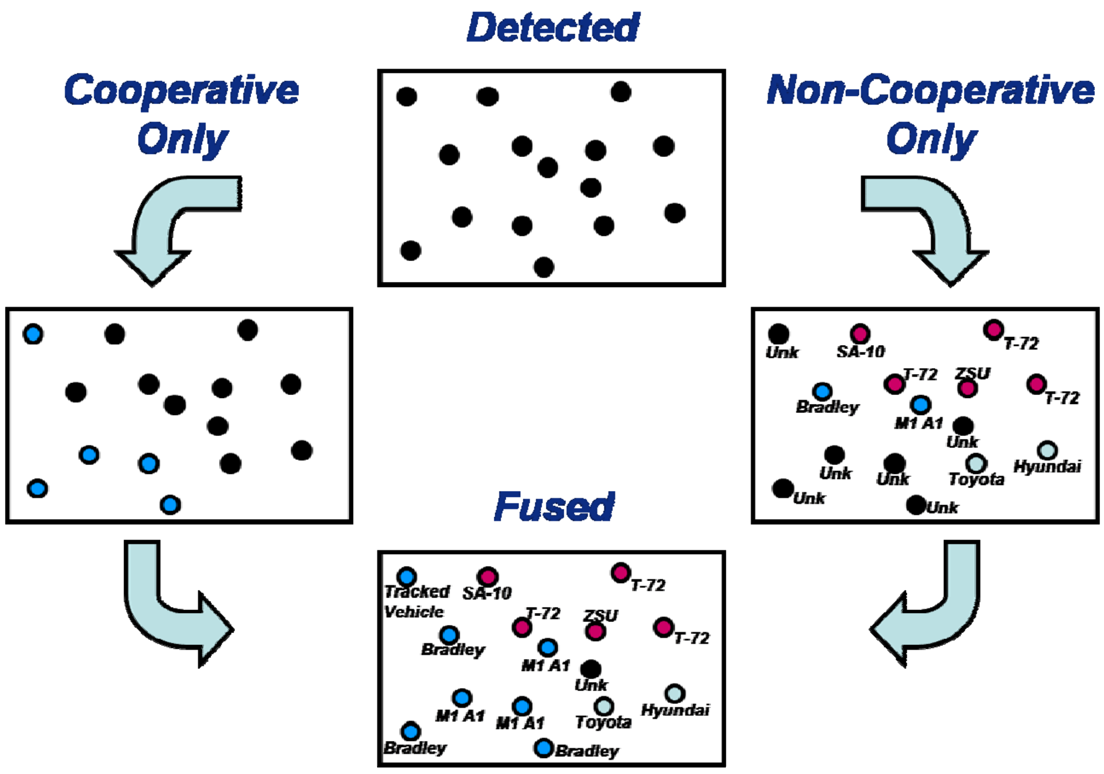


Figure 4: Characterizing CID Process [7]

Different Kinds of CID Scenarios

Air to Ground (A/G) [8]

An A/G CID architecture is a conceptual System of Systems capability that accurately characterizes entities in a combatant's area of responsibility to the extent that high confidence, real-time application of tactical options and weapon resources can occur. To accurately characterize entities, the A/G CID architecture is comprised of three types of platforms: a Forward Air-Controller (FAC), A/G shooters, and A/G surveillance. The FAC is assumed to be a ground FAC, the A/G shooter is a fixed-wing (FW) aircraft, and the A/G surveillance is provided by an Off-board or Unmanned Aerial Vehicle (UAV). Each platform can incorporate Cooperative and non-cooperative systems to identify ground targets. To pass information between all nodes, the CID architecture requires data networks. A fusion algorithm is also an important element of the CID architecture because it combines ID information from various sources into a single ID declaration. This architecture is depicted in figure 5.

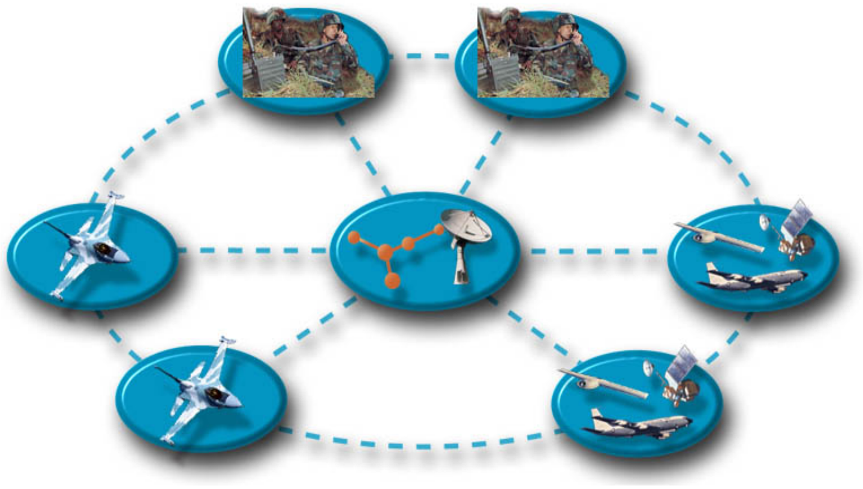


Figure 5: A/G CID Architecture [9]

A typical A/G target ID scenario is comprised of a complex picture of the battlespace where there exists lots of clutter, and as such, the set of ID possibilities are immense. Due to the infancy of the problem, the analysis tools that currently exist are immature. There are also no well-deployed Cooperative ID systems and there are virtually no non-cooperative ID systems. In a typical A/G mission, the warfighter knows (to some level) the target ID and the approximate target location. Additionally, the warfighter is loaded with the appropriate munitions and is knowledgeable in the rules of engagement (ROE) [10]. The warfighter can safely accomplish his/her mission if timely and accurate CID information is provided. The highest priority of the ground target issues that are of interest in CID are locating, detecting, and correlating an already identified hostile target. Cooperative ID solutions provide limited utility and will only help in certain missions such as Combat Air Support (CAS), Combat Search and Rescue (CSAR), and Special Operations missions, but these missions amount to less than 15% of all Air Force (AF) missions flown [10]. Also, Cooperative ID can only identify some friends, but cannot identify the enemy or neutrals. The value of non-cooperative ID is that it can provide target 'characterizations' needed to accomplish any mission. Unlike Cooperative ID, non-cooperative ID can identify friends, enemies, and neutrals.

Air to Air (A/A) [8]

Figure 6: Typical A/A target ID Scenario [10]

A typical A/A target ID scenario is comprised of a cleaner picture of the battlespace where there is less background clutter. Unlike the A/G scenario, the set of ID possibilities are significantly smaller. The A/A CID scenario has long been in the minds of the decision makers and thus has relatively mature analysis tools. In the A/A scenario, Cooperative ID technology forms the foundation of a family of systems; there exist numerous non-cooperative systems. One of the air target issues that are being considered is the use of off-board ID information. This has proven to be the toughest challenge when attempting to use off-board ID information for weapons/tactical decisions. The use of off-board ID information requires high positional accuracy due to correlation issues and data latency must be provided within a few seconds of the event it is needed. The second A/A issue is improving Cooperative ID. Just because Cooperative ID is improved, this does not directly equate to dramatically improved mission effectiveness. A slight mission effectiveness benefit is gained only when almost all friendly air targets possess and utilize the Cooperative ID system. The next issue is improving non-cooperative ID, this dramatically increases mission effectiveness. It helps warfighter make launch decisions at desired employment ranges if performance conditions (ranges, declaration rates, confidence rates) are good enough. Finally, preventing losses and preventing fratricide are other A/A key issues always in consideration. Non-cooperative systems do as much, if not more, than cooperative systems.

Modeling Architecture

Type of Modeling [4: Sec I, 6]

In a typical combat model the user must initialize the input variables of the model, execute the model to obtain output and analyze the output. There are three basic approaches to gathering a solution from a symbolic model: the analytic, numeric, and simulation model approaches. When one constructs a new model, the desired model approach will have a major impact on the structure of the resulting model.

Analytic Model

An analytic model consists of an explicit mathematical formula for each of the output variables written as a function of only the input variables [4: Sec I, 6]. Analytic models are desirable because the relationship between inputs and outputs is displayed as an explicit and hopefully simple formula [4: Sec I, 6]. Since the model solution is an explicit algebraic formula, the sensitivity of the model outputs to input variations can be analyzed symbolically by taking partial derivatives [4: Sec I, 6].

The article, Quantifying “Persistence” in the Context of Find-Fix-Finish (FFF), which is written by Roy E. Rice would be an example of an analytic model of CID modeling. The concept of "persistence" is a powerful enabler in ongoing operations, particularly in the context of a series of events often described as the "kill chain"; that is "find-fix-finish"(FFF) [11]. The author states that analysts are asked to show which part of the "kill chain" should be invested in more than the others. As an example, if one country invests in a new ISR (Intelligence, Surveillance, and Reconnaissance) capability, his ability to "find" targets will be improved. However, if the country invests in a new hypersonic munition, the time required to "finish" targets will be faster than before. Due to above reasons the author thought an analytic construct is needed to examine trade spaces. He first derived an equation for the probability of accomplishing the kill chain – P_{FFF}(T) then he quantified the drivers of this probability. He also added the development of "Persistence ratio" to quantify the relative impacts of persistent ISR, some results, and insights. The general expression for P_{FFF}(T) is

$$P_{FFF}(T) = 1 - \sum_j \frac{\lfloor e^{-T/\theta} \rfloor}{\prod_{i \neq j} (\theta_j - \theta_i)} \text{[11].} \tag{2.1}$$

T = time window, θ_i = mean-time-to event

As one can see, the model is described as an explicit formula.

Numeric Model [4: Sec I, 6-7]

A numeric model solution of an analytic model is obtained by first assigning numeric values to the input variables and then using the rules of mathematics to solve for numeric values of the output variables. Numeric model are possible for some models that cannot be solved in analytic closed form or where analytic models are possible but the output solutions are very complicated. Numeric model outputs restrictively allow analysts to estimate the model's response to other input scenarios. The partial derivatives of the model outputs can be estimated by numeric finite difference approximations. A good instance of such a model for which numeric solutions are beneficial is a linear program. Analytic formulas can show some of the characteristics of linear programming.

Simulation Model [4: Sec I, 7]

A simulation model solution is obtained by sequentially acting out the processes and interactions of the model. Simulation models are particularly appropriate for models whose relationships are expressed procedurally instead of algebraically. Simulation is the solution method that can best deal with complex, dynamic, high resolution models of force-on-force combat.

The output from a deterministic simulation model that does not contain probabilities or random effects will have the same form as the output from a numeric solution – a numeric value for each output variable. Sensitivity analysis or alternate scenarios will require that the simulation be replicated with some of the inputs changed.

The output from a stochastic simulation model which incorporates uncertain occurrences using probability distributions will be a single realization of the simulation as result of random variables. Such an output can be viewed as one possible battle result for the given input scenario. In order to understand the combat system in a scenario, it is necessary to replicate the simulation computer run with identical inputs but different random numbers used in the simulation sampling. Thus, several possible battle results are accumulated for each scenario, and from them analysts can approximate the distribution of the output measures or estimate mean values and confidence intervals. The primary model used in this research effort, CID is stochastic simulation model.

Simulation

Every simulation model including CID has its purpose: to describe the essential characteristic of the system as simply as possible. A replica of the real system may be the best representation. However, this is not the true purpose of the simulation model. In fact, most of the simulation model uses input/output products of the system to estimate or calculate the measure of performance. Entities, agents and objects similar to those in the actual world are used to estimate the performance easily.

Entities

The elements that make up the system are often referred to as entities [12:2]. In the frequently used war-game scenario, there are several hundred of these entities, but that is not complicated if all entities share similar program logic; particularly direct fire combat [4: Sec II, 8]. In CID, entities, including aircraft and other friendly sensors try to detect or classify objects as enemy forces on the battlefield. Enemy forces such as enemy aircraft, vehicles, and so on can be thought of as the entities. If the modeler creates an aircraft as an entity, its several on-board sensors can be attributes of that particular entity, the aircraft. In this effort, we are going to think about a small number of aircraft with different probabilities of successful detection and classification.

Agents

An agent is an active entity that has the capability of the component to make independent decisions [13:2]. Agents have certain characteristics:

- An agent is identifiable: a discrete individual with a set of characteristics and rules governing its behaviors and decision-making capability. Agents are self-contained. The discreteness requirement implies that an agent has a boundary and one can easily determine whether something is part of an agent, is not part of an agent, or is a shared characteristic.

- An agent is situated, living in an environment with which it interacts along with other agents. Agents have protocols for interaction with other agents, such as for communication, and the capability to respond to the environment. Agents have the ability to recognize and distinguish the traits of other agents.

- An agent may be goal-directed, having goals to achieve (not necessarily objectives to maximize) with respect to its behaviors. This allows an agent to compare the outcome of its behavior relative to its goals.

- An agent is flexible, having the ability to learn and adapt its behaviors based on experience. This requires some form of memory. An agent may have rules that modify its rules of behavior [13:3].

As the model is gets more complex and larger, the modeler needs to think about agent-based modeling. In CID, the reporting of a detection of enemy forces by friendly aircraft or doing an automatic classification process right after the detection may make an entity capable of independent decisions.

Objects

An object is a discrete item that can be selected and maneuvered, such as an onscreen graphic [14]. Usually, objects are data and the procedures to manipulate the data in object-oriented programming. An object-oriented simulation is focused on characterizing the manipulable data items (entities, objects) such that they are able to perform operations on and for themselves. It also focuses system implementation by the interaction of the data items. In the object-oriented approach, the model builder needs to send a message to invoke an operation on an object.

Receiver Operating Characteristics Curve

Receiver Operating Characteristics (ROC) analysis has been used to describe the tradeoff between true positive rate (TPR) and false positive rate (FPR) in signal detection theory. Besides being a commonly useful performance measure, ROC analysis has specific usefulness for skewed class distribution and different classification error costs. These properties are very important into the area of cost-sensitive learning and learning in the presence of unbalanced classes [15:1].

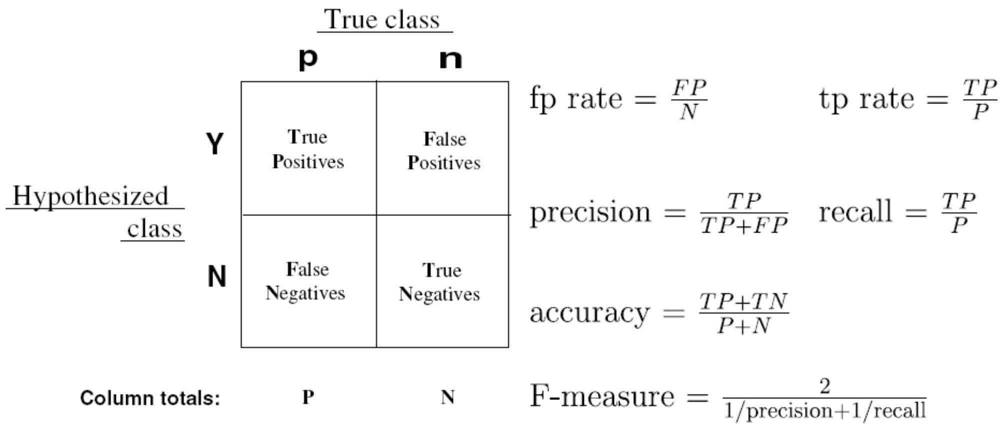


Figure 7: Confusion Matrix and Common Performance Metrics [15:2]

Figure 7 shows four possible outcomes based on classifier and instance. If the true class is positive and its simulation output is also positive, it is a true positive; if its predicted output is negative it is a false negative. If the true class is negative and its simulation output classified as negative, it is a true negative; if its predicted output classified as positive, it is a false positive. A set of true classes and predicted classes can be used to construct a two-by-two confusion matrix (CM).

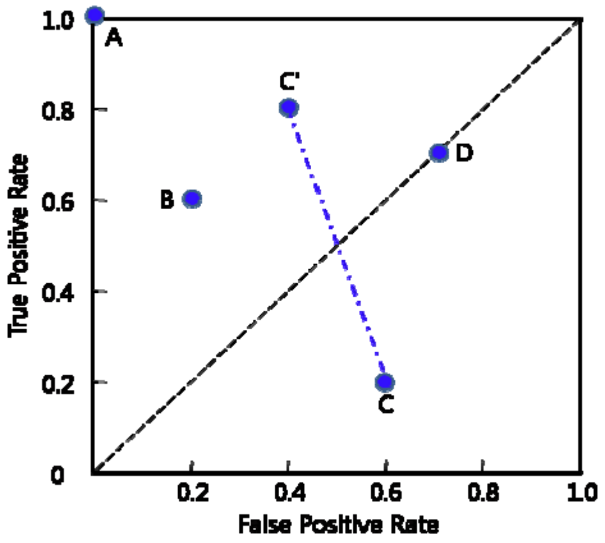


Figure 8: ROC Space Graph

ROC graphs have two dimensions in which the Y axis is true positive (TP) rate and the X axis is false positive (FP) rate. Figure 8 shows ROC space with five discrete classifiers generating a (FP rate, TP rate) pair corresponding to its class value. Point A, (0, 1) represents perfect positive classification. This point is the best possible prediction, representing 100% sensitivity (recall) and specificity (1-*fp* rate). If the performance is close to northwest (FP low, TP high), it means better classification. Classifiers appearing on the left-hand side of the ROC graph, near the X axis, may be thought of as "conservative": they make positive classifications only with strong evidence so they make few false positive errors, but they often have low TPRs as well [15:3]. Classifiers on the upper right-hand side of an ROC graph may be thought of as "liberal": they make positive classifications with weak evidence so they classify nearly all positives correctly, but they often have high FPRs [15:3]. In figure 8, B is more conservative than C'.

The point D on the diagonal line provides completely random guess. And the point C located in the lower right triangle shows worse performance than random guess. The relation between point C and C' shows an opposite condition of classification output on every true class – its TP rate becomes false negative rate (FNR) and its FP rate becomes true negative rate (TNR). Hence, point C in the lower right triangle is negated to point C' in the upper left triangle.

Radial Basis Functions

To make ROC curve approximations for this research we employ Radial Basis Functions (RBFs), which are a class of artificial neural networks with the ability to approximate non-linear functions and to interpolate data. A RBF ϕ has a symmetric output around a center μ, a sample site. That is, $\phi(x) = \phi(\|x - \mu\|)$, where the argument of ϕ is a vector norm, normally with $1 \le p \le 2$ (in this research the Euclidean-norm). A set of RBFs serves as a basis for representing multiple functions expressible as linear combinations of chosen RBFs and a polynomial basis *p(x)*:

$$y(x) = p(x) + \sum_{j=1}^{m} w_j \phi(\|x - x_j\|) \tag{2.2}$$

To solve for the weights *w* and polynomial coefficients *c* (note this *c* vector is different than the scalar in the RBF), we seek weights such that for interpolation values $f = (f_1, ..., f_m)$, $y(x) = f(x)$. Furthermore, because there are more parameters than data, we seek:

$$\sum_{j=1}^{m} w_j p(x_j) = 0 \tag{2.3}$$

This gives the following system,

$$\begin{pmatrix} A & P \\ P^T & 0 \end{pmatrix} \begin{pmatrix} w \\ c \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix} \tag{2.4}$$

where $A_{i,j} = \phi(\|x_i - x_j\|)$ for *i,j=1,…,m*, and $P_{i,j} = p_j(x_i)$ for *i=1,…n* and *j=1,…,k*, where *k* is the number of polynomial coefficients in the basis representation.

Several RBFs exist with various advantages to each. The bi-harmonic, $\phi(r) = r$ with a linear polynomial, and the tri-harmonic or cubic spline, $\phi(r) = r^3$ with a quadratic polynomial, are popular for fitting functions of three variables. The multi-quadric, $\phi(r) = \sqrt{r^2 + c^2}$, is useful for fitting topographical data. The thin-plate spline, $\phi(r) = r^2 \log(r)$, is popular for fitting smooth functions of two variables. Other RBFs include the inverse quadric, $\phi(r) = (r^2 + c^2)^{-1/2}$, and the Gaussian, $\phi(r) = \exp(-cr^2)$. Recall, *r* is the norm from equation (2.2). The parameter *c* is a positive constant, recommended to be equal to the mean of the distances between sample sites [16]. MATLAB includes a thin-plate spline in its Spline Toolbox, but it can only be used for functions of two variables. In this research, the thin-plate spline kernel and a linear base polynomial are used to generate the ROC curves since this combination made the most appropriate approximation throughout many trials. It should be mentioned that RBFs are much more general than presented here. In fact, the following are the true general definitions of classes of RBFs: 1) surface splines are any RBF such that $\phi(r) = r^k$, $k \in \mathbb{N}$ and odd, or $\phi(r) = r^k \log(r)$, $k \in \mathbb{N}$ and even; 2) multiquadrics are any RBF such that $\phi(r) = \sqrt{r^2 + c^2}^{\,k}$, $k > 0$ and $k \notin \mathbb{N}$; 3) inverse multiquadrics are any RBF such that $\phi(r) = \sqrt{r^2 + c^2}^{\,k}$ and $k < 0$; 4) Gaussians are any RBF such that $\phi(r) = \exp(-cr^2)$.

Monte Carlo Simulation

A Monte Carlo simulation has a algorithm that depends on repeated random numbers, U(0,1) random variates to compute its results. It is employed for finding solutions of certain stochastic or deterministic problems [25]. Due to its reliance on repeated calculation of random numbers, Monte Carlo simulation is appropriate applied in a computer, especially the MATLAB tool is exploited in this thesis. The name “Monte Carlo” simulation initiated during World War II, when this method was applied to problems associated with the development of the atomic bomb. Monte Carlo simulation has been widely applied to work out certain problems in statistics that are not analytically tractable [25:74].

Mathematical Framework to Optimize CID System

A mathematical framework for optimizing an ATR system is provided by Laine and Bauer [17]. Here we give a brief introduction concerning their work and subsequently exploit their efforts in this research.

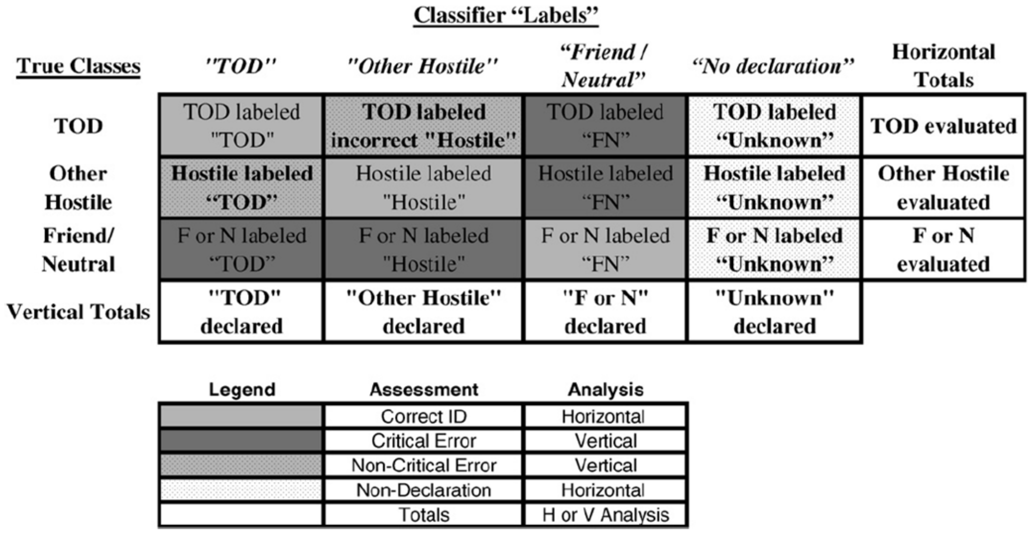
Summary of Previous Mathematical Frame Work for ATR

The purpose of their effort is to present the concepts of a new methodology to optimize ROC and rejection thresholds, for an ATR system [17:1]. They create an objective function that includes time and the constraints satisfy both decision maker preferences and traditional engineering measures of performance. Many CID problems including ATR can be modeled using two classes. For instance, an ATR system may declare an unknown object as “enemy” or “friend” in a situation that may include clutter and neutral forces [17:2]. ROC curve analysis is a well-known method to study CID systems. Given a finite data set, a standard ROC curve, f, can be thought of as a function of estimated performance measures [18]. They generated a ROC curve, *f*, by varying a decision threshold, θ, over its range, Θ as shown in

$$f = f(\theta) = \left\{ P_{FP}(\theta), P_{TP}(\theta) \mid \theta \in \Theta \right\} . \tag{2.5}$$

They introduced the CM below with respect to their true classes and classifier labels.

Table 1: Confusion Matrix with Two Hostile Classes and Rejection [17:5]



Classifier "Labels"

True Classes	***"TOD"***	***"Other Hostile"***	***"Friend / Neutral"***	***"No declaration"***	**Horizontal Totals**
TOD	TOD labeled "TOD"	**TOD labeled incorrect "Hostile"**	TOD labeled "FN"	**TOD labeled "Unknown"**	**TOD evaluated**
Other Hostile	**Hostile labeled "TOD"**	Hostile labeled "Hostile"	Hostile labeled "FN"	**Hostile labeled "Unknown"**	**Other Hostile evaluated**
Friend/ Neutral	F or N labeled "TOD"	F or N labeled "Hostile"	F or N labeled "FN"	**F or N labeled "Unknown"**	**F or N evaluated**
Vertical Totals	**"TOD" declared**	**"Other Hostile" declared**	**"F or N" declared**	**"Unknown" declared**	

Legend	**Assessment**	**Analysis**
▒	Correct ID	Horizontal
■	Critical Error	Vertical
▨	Non-Critical Error	Vertical
░	Non-Declaration	Horizontal
□	Totals	H or V Analysis

TOD: Target of the day, OH: Other hostile, FN: Friend or neutral, ND: No declaration

E_{CR}: Critical error, E_{NC}: Non critical error

They included a "non-declaration" option to their CM and did CM analysis using vertical and horizontal outputs. TOD is the most desired enemy target among a variety of enemies. To make mathematical formulation they deducted the probability of critical error is defined as probability associated with the union of the four output label and true class intersections, given a declaration is made, as shown in equation. (2.3) and (2.4) [17:5].

$$P(E_{\mathrm{CR}}) = P\left(\left(\begin{array}{c} P(''TOD''\cap FN)\cup P(''OH''\cap FN) \\ \cup P(''FN''\cap TOD)\cup P(''FN''\cap OH) \end{array}\right) \middle| declaration\right) \qquad (2.6)$$

$$P(E_{\mathrm{CR}}) = P\frac{\left(\begin{array}{c} P(''TOD''|FN)P(FN)+P(''OH''|FN)P(FN) \\ +P(''FN''|OH)P(OH)+P(''FN''|TOD)P(TOD) \end{array}\right)}{1-(P(''ND''|TOD)P(TOD)+P(''ND''|OH)P(OH)+P(''ND''|FN)P(FN))} \qquad (2.7)$$

They also derived non-critical errors (E_{NC}) in a similar manner to E_{CR}. Both E_{NC} and E_{CR} may be calculated directly from the estimated probabilities by the horizontal analysis of the CM frequency counts [17:6]. They classified their decision variables as discrete and continuous depending on characteristic of each variable. The decision variables may be discrete, such as an indicator variable to identify a specific fusion algorithm (F_i), use of specific sensors (S_j), or a predetermined number of minimum looks (*ML*), or they may be continuous, such as the thresholds associated with a sensor/fusion combination [17:6]. The objective function which is shown as equation (2.8), maximizing the probability of true positive target declarations across time, satisfied the determination of the optimal ATR system identified by the optimal fusion rule, with the optimal selection of: sensors, forced looks and a continuous threshold [17:7].

Objective function:

$$\arg\max_{x \in X} TRP(x) = \frac{P_{TP}(x)}{E(time_{TP}(x))} \quad \text{maximize } TPR(x), \text{ the true positive rate} \tag{2.8}$$

They also made several constraints, such as initial warfighter operational constraints, lower, fusion rule constraint, sensor selection constraints and minimum look constraint. Particularly, the warfighter operational constraints are limitations on incorrect fire decisions (vertical analysis), limitation of lower impact incorrect decisions (vertical analysis) and limitation of non-declarations (horizontal analysis). The remaining research consisted of experiments/results using the formulation. They also accomplished sensitivity analysis using different fusion algorithms.

Mathematical Frame Work for CID Simulation

The ATR system is a part of CID system. Particularly, an ATR system mainly deals with the target classification process. In CID both the detection and classification systems are very important and essential parts.

Table 2: Detection, Classification and System Confusion Matrices

	True Classes		
Detector "Labels"	**Enemy or Friend**	**Clutter**	**Horizontal Totals**
"Enemy or Friend"	E or F labeled "EF"	C labeled "EF"	**"E or F" Declared**
"Clutter"	E or F labeled "C"	C labeled "C"	**"C" Declared**
Vertical value	**E or F evaluated**	**Clutter evaluated**	

		True Classes		
Classifier "Labels"	**Enemy**	**Friend**	**Clutter**	**Horizontal Totals**
"Enemy"	E labeled "E"	F labeled "E"	C labeled "E"	**"E" Declared**
"Friend"	E labeled "F"	F labeled "F"	C labeled "F"	**"F" Declared**
Vertical value	**Enemy evaluated**	**Friend evaluated**	**Clutter evaluated**	

		True Classes		
System "Labels"	**Enemy**	**Friend**	**Clutter**	**Horizontal Totals**
"Enemy"	E labeled "E"	F labeled "E"	C labeled "E"	**"E" Declared**
"Friend"	E labeled "F"	F labeled "F"	C labeled "F"	**"F" Declared**
"Clutter"	E labeled "C"	F labeled "C"	C labeled "C"	**"C" Declared**
Vertical value	**Enemy evaluated**	**Friend evaluated**	**Clutter evaluated**	

The above three tables show a CM of the detection process (upper), that of classification process (middle) and that of system (lower). The color of each cell between the three tables represents relationships between the cells. This is because the classification process depends on the output of the detection process, that is, something has to be detected before proceeding into the classification process. We can see all the detected simulation output of the detection process at the classification CM. The sum of same colors at the system will coincide with the count shown at the detection process CM table of same color. If we try to calculate TPR and E_{CR} (critical error), we can use same idea presented in Laine and Bauer [17].

The TPR about enemy is $P("E"|E)$, the probability of enemy evaluation given true enemy. The expansion about this probability is shown in

$$P("E"|E) = \frac{P("E" \cap E)}{P(E)} \approx \frac{\text{first row and column of system's CM}}{\text{sum of first column of system's CM}} \tag{2.9}$$

As you see, TPR is described in vertical analysis of the CM frequency counts. The E_{CR}, the probability true friend given evaluation of enemy for fratricide, can be expressed in the similar manner.

$$P(F|"E") = \frac{P(F \cap "E")}{P("E")} \approx \frac{\text{first row and second column of system's CM}}{\text{sum of first row of system's CM}} \tag{2.10}$$

The E_{CR} is represented in horizontal analyses of the CM frequency counts. In this effort, we also define the label accuracy that is actually needed to a warfighter before he makes fire decision.

$$P(E|"E") = \frac{P(E \cap "E")}{P("E")} \approx \frac{\text{first row and column of system's CM}}{\text{sum of first row of system's CM}} \tag{2.11}$$

How CID modeling is currently performed in Combat Models

Background of Target Acquisition Process

CID is the most essential part of the current target acquisition process and has been developed dramatically in recent years. Most of target acquisition modeling is based on wartime work “Search and Screening”, report 56 of the U.S. Navy Operations Evaluation Group [4: Sec IV, 1]. In the report, Koopman defined detection as, “that event constituted by the observer's becoming aware of the presence and possibly of the position and even in some cases the motion of the target" [19]. The word, "detection" is used both generically to denote the entire target acquisition phenomenon (Koopman's definition) and as a particular level of acquisition in the above spectrum [4: Sec IV , 1]. If there is no clear description the intended meaning is specified in this section. From the OEG 56 report [19], two fundamental stochastic detection models, glimpse model and continuous looking model were established. Most subsequent detection model have included aspects of one or the other of Koopman's basic models [4: Sec IV, 2].

The Glimpse Model Combat Simulations

If the observer is searching one specific target, he will search the location of a target once per pass. Similarly, rotating scan radar will have its beam on the target once during each rotation [4: Sec Ⅳ, 2]. Such an intermittent opportunity of detection is a glimpse. Each glimpse considered to be a Bernoulli trial has g_i which is the probability of detection of the target on the i^{th} glimpse. And applying simple probability considerations, assuming independence of trial, yield

$$\text{Prob (no detect on first n-1 glimpse)} = \prod_{i=1}^{n-1}(1-g_i) \tag{2.12}$$

$$\text{Prob (first detect on n}^{\text{th}}\text{ glimpse)} = g_n\prod_{i=1}^{n-1}(1-g_i) \tag{2.13}$$

$$\text{Prob (detect in first } n \text{ glimpse)} = 1-\prod_{i=1}^{n}(1-g_i) \tag{2.14}$$

Where all the g_i s are equal the number of glimpses until first detection is distributed according to the geometric probability distribution [4: Sec IV, 3].

There glimpse probabilities, g_i are known for the prevailing observation conditions, and modelers want to use the glimpse model as the target acquisition module of a combat simulation. For each time step, the detection model can operate as follows.

```
ROUTINE TGT_ACQUISITION (OBS)
{Called for each observer during each time step}
    LOOP over each potential target, TGT
        Compute the value of $g_i$ as a function of observation conditions for this TGT.
        Note that $g_i$ = 0 is a possibility if the target is impossible to detect at this time.
        Simulate a Bernoulli trial with success probability = $g_i$.
        IF result is success then TGT is detected by OBS so add to detected list.
        END IF
    END LOOP
END ROUTINE [4: Sec IV, 4]
```

The above detection model considers a fixed time step and the time step interval is the same as the glimpse interval.

The Continuous Looking Model in Combat Simulations

The second fundamental stochastic detection model is the continuous looking model and has been used in most modern high resolution combat simulations due to its high flexibility and easy fitting into the structure of an event scheduled simulation [4: Sec IV, 6]. The continuous looking model has a detection rate function, D(t), which has the characteristic that the probability of detection in a short time is proportional to the length, Δ_T, of the interval and is given by:

$$\text{Prob (detect in [t, t+}\Delta_T\text{]) = D(t)}*\Delta_T \text{ [4: Sec IV, 6].} \tag{2.15}$$

If we assume constant D that D(t) = D for all t and use T = $n*(\Delta_T)$ in a longer time interval of length, we have the mathematics of the continuous looking model to yield:

$$\text{Prob (detect in length T) = 1 – Prob (fail n times) = 1 – (1 – D}*\Delta_T)^n$$

$$= 1\text{- (1 – } {DT}/{n})^n \text{ [4: Sec IV, 6].} \tag{2.16}$$

In the limit, n approaches infinity and Δ_T approaches zero with the constant T. This is identical to

$$\text{Prob (detect in length T) = 1 – } e^{(-DT)} \text{ [4: Sec IV, 6].} \tag{2.17}$$

The last formula is the cumulative distribution function of the exponential distribution and is very frequently used to model the time required to detect a target [4: Sec IV, 6]. In effect, the mathematics of the two fundamental models has equivalence. This is because the exponential distribution is the continuous analogue of the discrete geometric distribution which appeared in the glimpse model [4: Sec IV, 6]. Both have the assumption of independence of successive increments of time and the memory-less property [4: Sec IV, 6].

Simulation Analysis Techniques

To evaluate the output data, modelers would employ several techniques of analysis, since it is more advisable than doing just one technique. If the modeler uses one technique, he may get an incorrect evaluation about the output data. In this effort, three different evaluation methods are contrasted.

Linearly Constrained Discrete Optimization (LCDO)

Optimization is an important tool in decision science and in the analysis of systems [21]. To make use of this tool, we have to first identify an objective function and its variables. And often, the variables are limited by some constraints. In this research, our objective is to find the threshold combination that minimizes the cost and maximizes label accuracies. The threshold combinations of this effort are drawn from ROC curves and they are members of a finite set. Before we optimize, we need to make an appropriate model that identifies the objective, variables, and constraints for a given problem [21]. The model description including variable definitions will be presented in the next Chapter.

Mathematically, optimization is the minimization or maximization of a function subject to constraints on its variables [21]. We generally use the following notation:

1. x is the vector of variables, also called parameters or unknowns;
2. f is the objective function, a (scalar) function of x that we want to maximize of minimize;
3. c_i are constraint functions, which are scalar functions of x that define certain equations and inequalities that the unknown vector x must satisfy [21].

Using this notation, the optimization problem can be written as follows:

$$\min_{x \in R^n} f(x) \quad \text{subject to} \quad \begin{array}{l} c_i(x) = 0,\ i \in E, \\ c_i(x) \geq 0,\ i \in I \end{array} \quad [21]. \tag{2.18}$$

Here I and E are sets of indices for equality and inequality constraints, respectively.

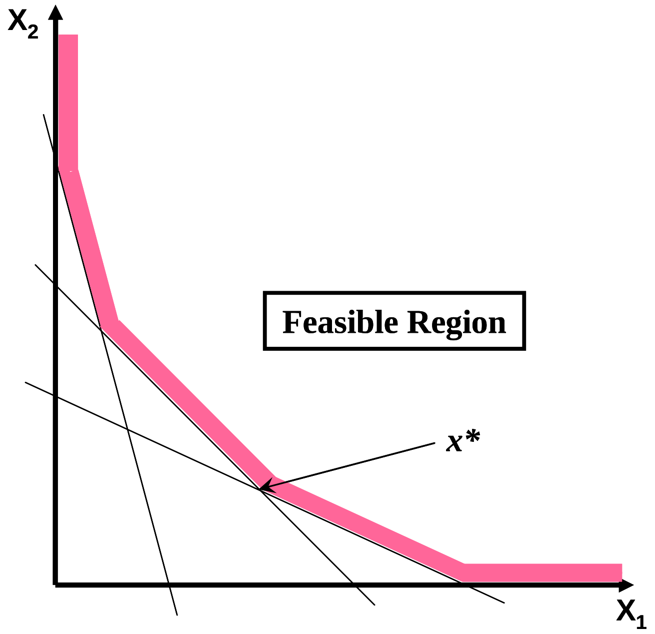


Figure 9: Example of Geometrical Representation of General Optimization Problem

The feasible region which shown in figure 9 is the set of points satisfying all the constraints, and the point x*, which is the solution of the problem. Sometimes it is necessary to label the variables with two or three subscripts.

Robust Parameter Design with Taguchi's S|N ratio: Crossed Array Design

The RPD is an approach to produce a realization of activities that emphasizes choosing the levels of controllable factors (or parameters) to accomplish two objectives: (1) to ensure that the mean of the output response is at a desired level or target and (2) to ensure that the variability around this target value is as small as possible [22:464]. The general RPD problem was developed by Dr Taguchi in 1960 and the Taguchi method has been widely used in the process of finding factors that are most important in achieving objectives in industrial processing [23]. The Taguchi method has an important aspect in terms of variability, caused by considering certain types of variables as noise variables or uncontrollable variables [22:466]. An essential part of the RPD problem is identifying controllable variables and the uncontrollable variables, noise variables that affect process or product performance, and then finding the optimal settings for the controllable variables that achieve an optimal objective function value while minimizing the variability transmitted from the noise variables [22:466].

Table 3: Example of Crossed Array Matrix [22: 468]

(a) Inner Array					(b) Outer Array									Responses	
					E	-	-	-	-	+	+	+	+		
					F	-	-	+	+	-	-	+	+		
					G	-	+	-	+	-	+	-	+		
Run	A	B	C	D										mean	SN_L
1	-1	-1	-1	-1		15.6	9.5	16.9	19.9	19.6	19.6	20.0	19.1	17.525	24.025
2	-1	0	0	0		15.0	16.2	19.4	19.2	19.7	19.8	24.2	21.9	19.425	25.522
3	-1	+1	+1	+1		16.3	16.7	19.1	15.6	22.6	18.2	23.3	20.4	19.025	25.335
4	0	-1	0	+1		18.3	17.4	18.9	18.6	21.0	18.9	23.2	24.7	20.125	25.904
5	0	0	+1	-1		19.7	18.6	19.4	25.1	25.6	21.4	27.5	25.3	22.825	26.908
6	0	+1	-1	0		16.2	16.3	20.0	19.8	14.7	19.6	22.5	24.7	19.225	25.326
7	+1	-1	+1	0		16.4	19.1	18.4	23.6	16.8	18.6	24.3	21.6	19.850	25.711
8	+1	0	-1	+1		14.2	15.6	15.1	16.8	17.8	19.6	23.2	24.2	18.313	24.852
9	+1	+1	0	-1		16.1	19.9	19.3	17.3	23.1	22.7	22.6	28.6	21.200	26.152

Taguchi proposed two statistics from the crossed array design: the average of each observation in the inner array composed of controllable variable combinations across all runs in the outer array composed of noise variable combinations, and a summary statistic about the mean and variance, called *signal-to-noise*(S|N) ratio [22:468]. Then an analysis is performed to find out which setting of the controllable factors form the mean as close as possible to the desired response and a maximum value of the S|N ratio [22:469].

These S|N ratios are derived from the quadratic loss function, and here are three functions that are widely used and applicable [24: 418].

- Nominal the best: $$SN_T = 10\log\left(\frac{\overline{y}^2}{S^2}\right) \tag{2.19}$$
- Larger the better: $$SN_L = -10\log\left(\frac{1}{n}\sum_{i=1}^{n}\frac{1}{y_i^2}\right) \tag{2.20}$$
- Smaller the better $$SN_S = -10\log\left(\frac{1}{n}\sum_{i=1}^{n}y_i^2\right) \tag{2.21}$$

However, this mean and variance modeling approach using a crossed array design may not explain appropriately the direct interactions between controllable variables and noise variables and in some examples, it can even mask these relationships [22:471]. If we care about the S|N$_s$ (smaller-the better) ratio, equation (2.21), maximizing S|N$_s$ will minimize $(\frac{1}{n}\sum_{i=1}^{n}y_i^2)$. However, it is easy to show that

$$(\frac{1}{n}\sum_{i=1}^{n}y_i^2) = \overline{y}^2 + \frac{1}{n}\left(\sum_{i=1}^{n}y_i^2 - n\overline{y}^2\right) = \overline{y}^2 + \left(\frac{n-1}{n}\right)S^2 . \tag{2.22}$$

Therefore, the use of S|N$_s$ as a response variable confounds location and dispersion effects [24: 429].

Robust Parameter Design: Combined Array Design and the Response Model

Table 4: Example of a Combined Array Matrix [22:476]

Run number	x1	x2	z1	z2	z3	y
1	-1.00	-1.00	-1.00	-1.00	1.00	44.2
2	1.00	-1.00	-1.00	-1.00	-1.00	30.0
3	-1.00	1.00	-1.00	-1.00	-1.00	30.0
4	1.00	1.00	-1.00	-1.00	1.00	35.4
5	-1.00	-1.00	1.00	-1.00	-1.00	49.8
6	1.00	-1.00	1.00	-1.00	1.00	36.3
7	-1.00	1.00	1.00	-1.00	1.00	41.3
8	1.00	1.00	1.00	-1.00	-1.00	31.4
9	-1.00	-1.00	-1.00	1.00	-1.00	43.5
10	1.00	-1.00	-1.00	1.00	1.00	36.1
11	-1.00	1.00	-1.00	1.00	1.00	22.7
12	1.00	1.00	-1.00	1.00	-1.00	16.0
13	-1.00	-1.00	1.00	1.00	1.00	43.2
14	1.00	-1.00	1.00	1.00	-1.00	30.3
15	-1.00	1.00	1.00	1.00	-1.00	30.1
16	1.00	1.00	1.00	1.00	1.00	39.2
17	-2.00	0.00	0.00	0.00	0.00	46.1
18	2.00	0.00	0.00	0.00	0.00	36.1
19	0.00	-2.00	0.00	0.00	0.00	47.4
20	0.00	2.00	0.00	0.00	0.00	31.5
21	0.00	0.00	0.00	0.00	0.00	30.8
22	0.00	0.00	0.00	0.00	0.00	30.7
23	0.00	0.00	0.00	0.00	0.00	31.0

Due to the weakness of the crossed array design, Montgomery [22] suggests combined array designs and the response model approach, which may be more appropriate for interactions between controllable variables and noise variables (rather than the crossed array design and its mean and variance approach). The model takes the familiar regression form of:

$$y = \beta_0 + \sum_{i=1}^{n} \beta_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \beta_{ij} x_i x_j + \sum_{i=1}^{r} \gamma_i z_i + \sum_{i=1}^{n} \sum_{j=1}^{r} \delta_{ij} x_i z_j + \varepsilon \qquad (2.23)$$

where β s are the control coefficients, γ s are the noise coefficients and δ s are the interaction coefficients. If we generalize the above regression result with k controllable variables and r noise variables, the general response model is

$$y\ (x,z) = f(x) + h(x,z) + \varepsilon \qquad (2.24)$$

$$f(x) = \beta_0 + \sum_{i=1}^{n} \beta_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \beta_{ij} x_i x_j \qquad (2.25)$$

$$h(x,z) = \sum_{i=1}^{r} \gamma_i z_i + \sum_{i=1}^{n} \sum_{j=1}^{r} \delta_{ij} x_i z_j \qquad (2.26)$$

where *f(x)* is the portion involving only the controllable variable and *h(x, z)* are the terms involving the main effects of the noise factors and interactions between the controllable and noise factors [22:472]. Because noise variables are coded such that the design points are symmetric about zero, the mean model for the response expectation is:

$$E_z[y(x,z)] = f(x) = \beta_0 + \sum_{i=1}^{n} \beta_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \beta_{ij} x_i x_j \text{ [22:473]} \qquad (2.27)$$

and the variance model is:

$$V_z[y(x,z)] = \sum_{i=1}^{r} \left[\frac{\partial y(x,z)}{\partial z_i} \right]^2 \sigma_{z_i}^2 + \sigma^2 \text{ [22:473].} \qquad (2.28)$$

Contour plots and surface plots are typically used to show mean model and variance model. The goal is to find the set of parameters that optimize the response with the minimal variance.

III. Methodology

Introduction

The CID scenario that this effort is considering is an Air to Ground scenario. The basic concept is shown in the figure below.

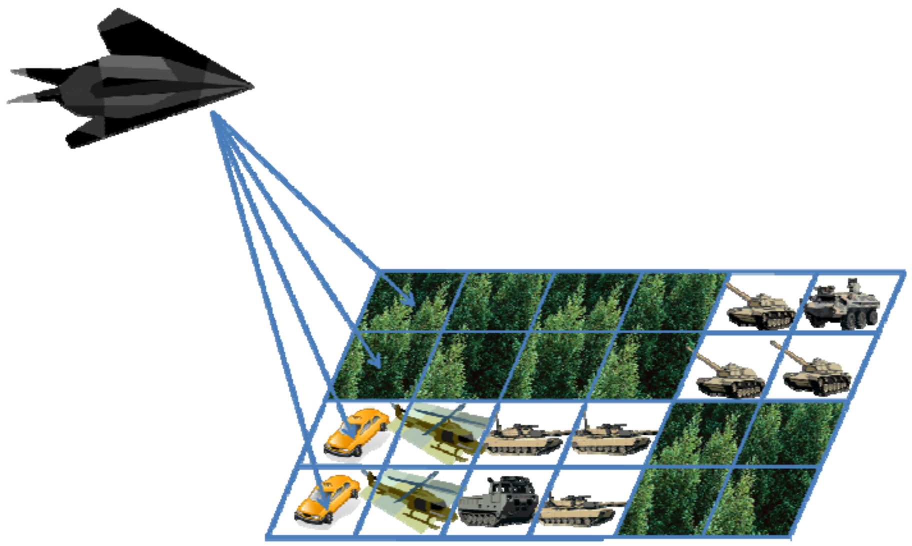
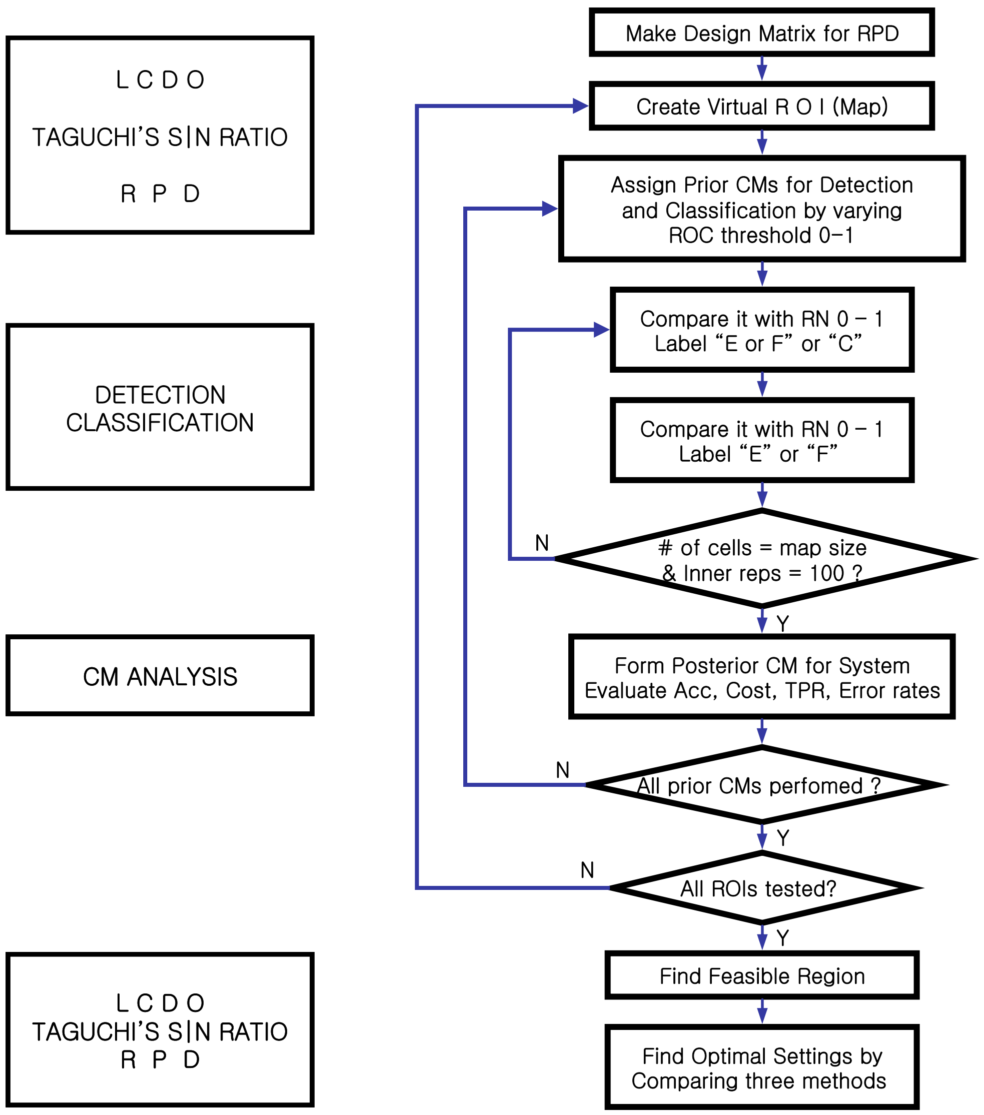


Figure 10: Concept Picture of CID Process

First, the friendly force's aircraft divide the ROI into constant size blocks. Then the aircraft performs detection and classification for each block and in the model save the result as data. In this effort, we assume Non-cooperative communication for doing detection and classification in the given ROI, and declare enemy, friend or clutter based on the output of the system. The ROC analysis and Monte Carlo simulation mentioned in Chapter 2 are used to create the responses (the label accuracy and the cost) of the simulation. After finding the responses, we can obtain optimal ROC threshold settings by applying a LCDO, Taguchi's S|N ratio method and RPD with a combined array design.

CID simulation needs several inputs such as: an artificially formed area (battlefield) consisting of enemies, friends, neutrals and clutter, prior confusion matrices obtained from predetermined ROC curves and cost coefficients associated with the incorrect detection and classification. In this research, the prior ROC threshold is identical to the prior CM because, predetermined ROC thresholds are expressed through the prior CM. The most important output data of the CID simulation is the CM with attributes to obtain optimal ROC thresholds settings which optimize objective functions such as maximum label accuracy of the system and minimum error cost. To create the posterior CM, we need a logical random simulation process such as Monte Carlo simulation to describe the actual detection and classification processes.

Description of Simulation model



L C D O
TAGUCHI'S S|N RATIO
R P D
DETECTION
CLASSIFICATION
CM ANALYSIS
L C D O
TAGUCHI'S S|N RATIO
R P D
Make Design Matrix for RPD
Create Virtual R O I (Map)
Assign Prior CMs for Detection and Classification by varying ROC threshold 0–1
Compare it with RN 0 – 1 Label "E or F" or "C"
Compare it with RN 0 – 1 Label "E" or "F"
# of cells = map size & Inner reps = 100 ?
N
Y
Form Posterior CM for System Evaluate Acc, Cost, TPR, Error rates
All prior CMs perfomed ?
N
Y
All ROIs tested?
N
Y
Find Feasible Region
Find Optimal Settings by Comparing three methods

Figure 11: Flowchart of CID Process

First, the model executes the design matrix for simulation, then it creates a virtual ROI and assigns prior CMs according to design matrix. After creating the enviornment of the system, the model repeats detection and classification processes appropriately and it deduces a posterior CM to obtain data needed when we analyze simulation performance. If the system tested all the prior CM combinations in the pre-created ROI, the model creates the next ROI and repeats what it has been done before. Unless the system has tested all the prior CM combinations, the model assigns next prior CM and completes its detection and classification processes until all the prior CM combinations are tested in the present ROI. After examining all simulation design points, the feasible region can be obtained by examinning several constraints. Then the optimal thresholds combination can be found by LCDO, Taguchi's S|N ratio method, and RPD with a combined array. Finally, a confirmation experiment is performed to compare each solution from different methods.

Making the Design Matrix

This simulation model has two responses: the label accuracy and the cost. To get optimal thresholds combination, we employ three methods: LCDO, Taguchi's S|N ratio and RPD. The LCDO procedure is relatively simpler than the others because, it just deduces the prior CMs that optimize the objective function by satisfying constraints. We need to make a design matrix according to design of experiment principles in order to employ RPD techniques. In the case of using Taguchi's S|N ratio, a crossed array design matrix is needed and in the case of using RPD, a combined array design matrix is required to perform the simulation. This model first uses a combined array design matrix and obtains simulation responses. Then the combined array matrix is transformed to a crossed array design matrix by easy MATLAB manipulation.

Table 5: Example of the Design Matrix

Comb #	TPR_D	FPR_D	TPR_C	FPR_C	Map size	# of Enemy	# of Friend	Cost Coef
1	0.4422	0.0005	0.4082	0.0005	100	5	5	1
2	0.4932	0.001	0.4082	0.0005	100	5	5	1
3	0.5694	0.0015	0.4082	0.0005	100	5	5	1
4	0.6098	0.002	0.4082	0.0005	100	5	5	1
5	0.644	0.0025	0.4082	0.0005	100	5	5	1
			-			-		
			-			-		
100828	0.9102	0.014	0.68	0.0045	100	40	5	2
100829	0.9292	0.0145	0.68	0.0045	100	40	5	2
100830	0.9307	0.015	0.68	0.0045	100	40	5	2
100831	0.9308	0.0155	0.68	0.0045	100	40	5	2
100832	0.9308	0.016	0.68	0.0045	100	40	5	2
			-			-		
			-			-		
159992	1	0.046	0.9667	0.05	1000	40	40	2
159993	1	0.0465	0.9667	0.05	1000	40	40	2
159994	1	0.047	0.9667	0.05	1000	40	40	2
159995	1	0.0475	0.9667	0.05	1000	40	40	2
159996	1	0.048	0.9667	0.05	1000	40	40	2
159997	1	0.0485	0.9667	0.05	1000	40	40	2
159998	1	0.049	0.9667	0.05	1000	40	40	2
159999	1	0.0495	0.9667	0.05	1000	40	40	2
160000	1	0.05	0.9667	0.05	1000	40	40	2

There are controllable factors and noise factors in the design matrix shown in table 5. The controllable factors are ROC thresholds for detection and classification and the noise factors are the size of ROI represented as the total sum of grid points, the number of enemy targets, the number of friendly targets and the cost coefficient for fratricide. We have two controllable factors with 100 levels each, and four noise factors with 2 levels each. Thus the number of design points is 160,000 and the whole forms a full factorial design ($100^2*2^4=160000$). In effect, if the objective function is the label accuracy, the cost coefficient is not necessary and the number of noise factor is reduced from four to three. Thus, we only need 80,000 design points because of $100^2*2^3=80000$.

However, here a design matrix is analyzed that has 160,000 design points to acquire two responses at the same time.

Establishment of Virtual ROI to Set up System Environment

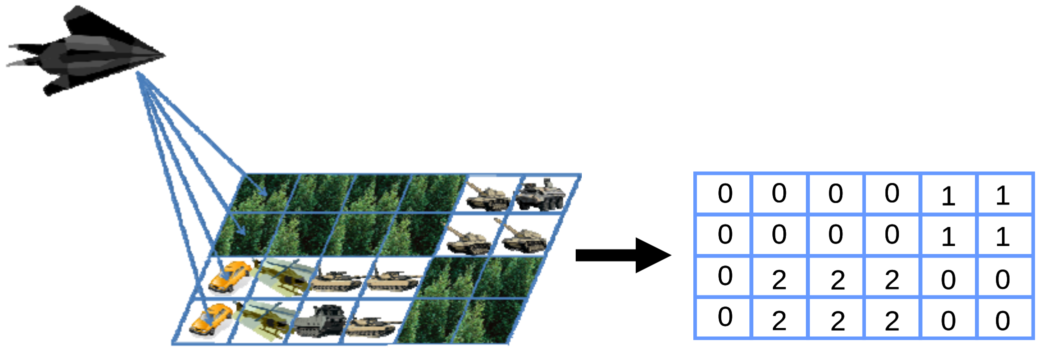


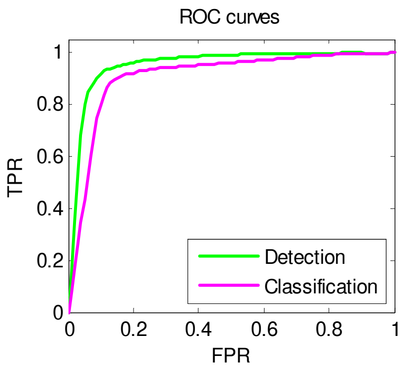
Figure 12: Configuration of ROI

Figure 12 shows the process of configuring a real ROI to virtual ROI via a matrix to execute simulation. The CID process needs a virtual ROI to employ given thresholds since detection and classification use a virtual ROI when they evaluate each grid with a specific prior ROC threshold. There are a number of components that construct the actual battlefield; however, this model deals typically with enemy, friend, and clutter (including neutral, civilian, and every object except enemy and friend). In the virtual ROI, the enemy is represented by"1", friend is represented by "2", and clutter is expressed by "0". Each grid point can only have one characteristic out of three (enemy, friend and clutter). As it is shown at figure 12, the matrix established by these three figures can be thought as a virtual ROI. Once the virtual ROI is established, the system tests all ROC threshold combinations by comparing it with random numbers and declares the grid point enemy, friend or clutter based on the result of the comparison. The virtual ROI is considered a noise factor because in the case of a real battlefield, the size of the

ROI, the characteristics of the grid (enemy, friend or clutter), the number of enemy and that of friend in the ROI, and so forth are generally hard to predict.

Assignment of Prior CM

After establishing a virtual ROI, the system assigns the prior CMs, ROC thresholds that are read from predetermined ROC curves for detection and classification. Each one hundred (FPR, TPR) pair for the detection and the classification processes is used in this simulation. The number of different combination of detection and classification (DEC (FPR, TPR), CLASS (FPR, TPR)) threshold is 10,000 due to 100*100 = 10000.



	True Classes	
Detector "Labels"	Enemy or Friend	Clutter
"Enemy or Friend"	Prob ("EF" | E or F)	Prob ("EF" | C)
"Clutter"	Prob ("C" | E or F)	Prob ("C" | C)

		True Classes	
Classifier "Labels"	Enemy	Friend	Clutter
"Enemy"	Prob ("E" | E)	Prob ("E" | F)	Prob ("E" | C) = 0.5
"Friend"	Prob ("F" | E)	Prob ("F" | F)	Prob ("F" | C) = 0.5

Figure 13: Description of Deduction of Prior CMs from ROC Curves

Figure 13 describes the process of constructing prior CMs, detection (upper) and classification (lower) from two different ROC curves. The detection ROC curve has a higher TPR at a same FPR (relatively low FPR) than that of the classification ROC curve since the detection is easier than the classification. That is, generally, the probability of correct detection is higher than that of correct classification. In a real CID simulation model, the simulator has to use ROC curve having a much lower FPR than that of FPR shown in figure 13.

In the upper-left hand corner of the detection prior CM (denoted the (1,1) block), the TPR of the detection is the probability of being detected at detection process when an object (enemy or friendly force) exists at the one particular block. In the (1, 2) block of the detection prior CM, the FPR of the detection is the probability being detected even though the block is clutter. In the (1, 3) block of the detection prior CM, the FNR of the detection is the probability of not being detected although there is an object (enemy or friend) in the block. In the (1, 4) block of the detection prior CM, the TNR of the detection is the probability of declaring clutter when the grid is clutter. The (1, 1) block and the (1, 4) block represent correct detection; however, the (1, 2) block and the (1, 3) block indicate errors of detection process and thus they will incur costs of themselves.

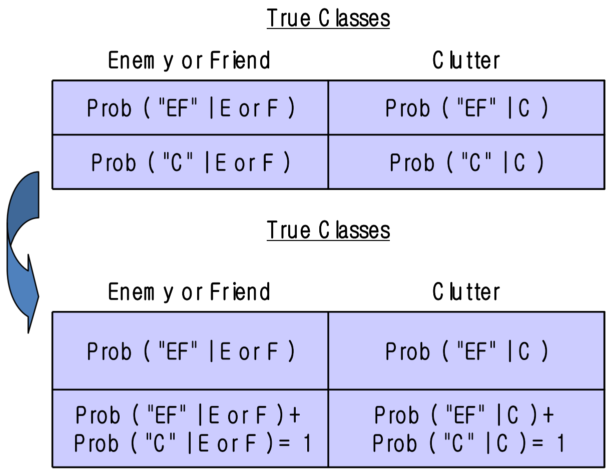
Classification is performed based on the result of the detection. In the case of declaring clutter as an object, the grid which is actually a clutter enters the classification process. Thus, there is one more column than the original prior classification CM which reflects a 50-50 percent probability to declare clutter incorrectly detected as objects. On the other hand, detection may be failed although the block has an object (enemy or a friendly force). In this case, the block which should have been detected can not go into classification process and later, it will be classified as detection error. The two detection errors which have been explained are very likely to happen in the real CID system owing to FPR and FNR of the detection. The first two rows and columns of the prior classification CM are based on correct detection about enemy or a friendly object.

In the upper-left hand corner of the classification prior CM (denoted the (1, 1) block), the TPR of the classification is the probability of being classified as enemy at classification process when the enemy actually exists at the one specific block and it has been detected through the detection process. In the (1, 2) block of the classification prior CM, the FPR of the classification is the probability of being classified as enemy even though the true class of detected object is a friendly force. In the (2, 1) block of the classification prior CM, the FNR of the classification is the probability of being classified as a friendly force although the true class of detected grid is enemy. In the (2, 2) block of the classification prior CM, the TNR of the classification is the probability of being classified as a friendly force when the true class of the grid is a friendly force and that particular grid has been detected through the detection process. After assigning these prior CMs, the system gets into the detection and classification processes to test those prior CMs.

Detection and Classification Processes

The model established a virtual ROI according to the design matrix at the opening of the simulation. The system performs detection and classification processes and makes posterior CMs by employing 10,000 prior CM combinations at the established virtual ROI. To test one prior CM combination, this study uses Monte Carlo simulation, a random number comparison method. That is, the system compares its prior CM combinations with a random number from 0 to 1 in terms of every grid point which is on the pre-established virtual ROI and decides success or failure of the detection and the classification.

```
for k = 2:numberchoices
   out(k,1) = prob(k) + out(k-1,1);
end
check = 0;
index = 1;
while check == 0
   if out(index,1) >= rand(1)
      output1(i,j) = column_d(index);
      check = 1;
   else
      index = index + 1;
   end
end
```



True Classes	
Enemy or Friend	Clutter
Prob ("EF" | E or F)	Prob ("EF" | C)
Prob ("C" | E or F)	Prob ("C" | C)

True Classes	
Enemy or Friend	Clutter
Prob ("EF" | E or F)	Prob ("EF" | C)
Prob ("EF" | E or F) + Prob ("C" | E or F) = 1	Prob ("EF" | C) + Prob ("C" | C) = 1

Figure 14: The Part of Detection and Classification MATLAB Code and its Description

As we see at ROC curve theory of Chapter 2, the sum of TPR and FNR and that of FPR and TNR are equal to 1. The right figure (a prior CM for detection) of the figure 14 explains a transition of the assigned prior CM and the first three lines of the left MATLAB code make these transition. The MATLAB function, "Rand (1)" creates random number between 0 and 1. For example, when there is an object (enemy or a friendly force at) on the one grid point of the established virtual ROI and the "Rand (1)" is equal to 0.623. Then if the TPR of detection is greater than 0.623, the process recognizes the detection of the object but if not greater than 0.623 the process declares that grid point as clutter. In case of detection, the situation always can be included within one of both mentioned cases because, "Rand (1)" is smaller than one and the sum of TPR and FNR is always one.

Detection is the process of determining whether there is an object (enemy or a friendly force) in the ROI or not. The meaning of having detected something is that there is enemy or a friendly object and that of having detected nothing is that there is nothing as unusual. This model uses integers to express events in the program. If the system detects something, the output of that particular grid point is "3" and if not the output is "0". The grid points with results of "3" at detection process automatically go to classification process.

The classification process declares enemy or a friendly object using a similar method to that of the detection process. However, as mentioned before, the system gives 50-50 probability for the incorrect detection about clutter and declares enemy or a friendly force by employing Monte Carlo simulation (random number comparison).

Due to using random numbers in Monte Carlo simulation, we need an appropriate number of inner replications so that we can estimate mean values. The system which is used in this effort sets up 100 replications because as shown figure 15 the constant variance state appears after one hundred inner replications.

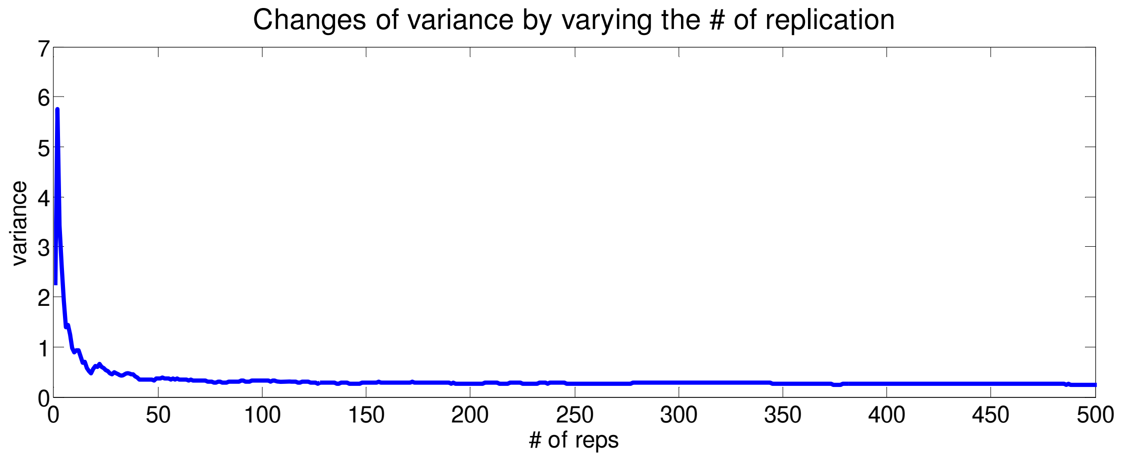


Figure 15: Changes of Variance by Varying the Number of Replications

Evaluating posterior CM for Analysis

After finishing the detection and classification processes with a specific prior CM combination, we begin our analysis. The analysis is performed by horizontal and vertical methods. The posterior CM is made by comparing system's declaration with the assigned virtual ROI. The system can do nine different system declarations because, the model assumes that there are only three characteristics (enemy, a friendly force or clutter) at the ROI. For example, in the case of assigning enemy to the first grid point of the virtual ROI, system declaration is one of three such as "enemy", "friend", or "clutter" and if it is declared as enemy, one count is added to the (1, 1) block of the system posterior CM.

Table 6: Detection, classification and system posterior CMs (counts)

	True Classes		
Detector "Labels"	Enemy or Friend	Clutter	Horizontal Totals
"Enemy or Friend"	E or F labeled "EF"	C labeled "EF"	"E or F" Declared
"Clutter"	E or F labeled "C"	C labeled "C"	"C" Declared
Vertical value	E or F evaluated	Clutter evaluated	

	True Classes			
Classifier "Labels"	Enemy	Friend	Clutter	Horizontal Totals
"Enemy"	E labeled "E"	F labeled "E"	C labeled "E"	"E" Declared
"Friend"	E labeled "F"	F labeled "F"	C labeled "F"	"F" Declared
Vertical value	Enemy evaluated	Friend evaluated	Clutter evaluated	

	True Classes			
System "Labels"	Enemy	Friend	Clutter	Horizontal Totals
"Enemy"	E labeled "E"	F labeled "E"	C labeled "E"	"E" Declared
"Friend"	E labeled "F"	F labeled "F"	C labeled "F"	"F" Declared
"Clutter"	E labeled "C"	F labeled "C"	C labeled "C"	"C" Declared
Vertical value	Enemy evaluated	Friend evaluated	Clutter evaluated	

We can see three different posterior CMs at table 6. The first table (upper) is the posterior CM of the detection process and the second one (middle) is that of the classification process and the third one (lower) is that of the system. The system posterior CM is mainly exploited for analysis because, it covers both the detection and classification posterior CM (The first two posterior CMs are included in the system one). The count of purple components of the detection posterior CM represents the number of successful results of the detection process associated with the virtual ROI and it is same as the sum of purple components of the classification posterior CM and those of the system posterior CM. The diagonal of a posterior system CM means the count of correct results of a system and the others (off-diagonals) represent the count of each possible error of a system. The total sum of counts of a posterior system CM is equal to that of grid points (size of virtual ROI).

The upper-left hand corner of the system posterior CM (denoted the (1, 1) block) is the TPR of the system. It is one of our constraints and it can be evaluated by following formula.

$$system\ TPR\ =\ P("E"|E) = \frac{P("E"\bigcap E)}{P(E)} \approx \frac{\text{first row and column of system CM}}{\text{sum of first column of system CM}} \qquad (3.1)$$

System errors are classified into four categories according to their characteristics. The first one is the error that can cause fratricide, $E_{CR}^{1} = P(F\,|"E")$. The second one is the error of declaring enemy as a friendly force, $E_{CR}^{2} = P(E\,|"F")$. The third one is one of detection errors and it happens when the clutter is declared as an object (enemy or a friendly force), $E_{CR}^{3} = P(C\,|"E"or"F")$. The last one is occurred when the system fails to detect an object (enemy or a friendly force) and declares it as clutter, $E_{CR}^{4} = P(E \text{ or } F \,|\text{"C"})$. Throughout the horizontal analysis in terms of the system posterior CM, we can get following critical error rates.

$$E_{CR}^{1} = P(F\,|\text{"}E\text{"}) = \frac{P(F \cap \text{"}E\text{"})}{P(\text{"}E\text{"})} \approx \frac{\text{first row and second column of system CM}}{\text{sum of first row of system CM}} \tag{3.2}$$

$$E_{CR}^{2} = P(E\,|\text{"}F\text{"}) = \frac{P(E \cap \text{"}F\text{"})}{P(\text{"}F\text{"})} \approx \frac{\text{second row and first column of system CM}}{\text{sum of second row of system CM}} \tag{3.3}$$

$$E_{CR}^{3} = P(C\,|\text{"}E\text{"}or\text{"}F\text{"}) = \frac{P(C \cap \text{"}E\text{"}) + P(C \cap \text{"}F\text{"})}{P(\text{"}E\text{"}) + P(\text{"}F\text{"})}$$

$$\approx \frac{\text{sum of first and second rows at third column of system CM}}{\text{sum of first and second rows at all columns of system CM}} \tag{3.4}$$

$$E_{CR}^{4} = P(E \text{ or } F\,|\text{"C"}) = \frac{P(E \cap \text{"}C\text{"}) + P(F \cap \text{"}C\text{"})}{P(\text{"}C\text{"})}$$

$$\approx \frac{\text{sum of first and second columns at third row of system CM}}{\text{sum of third row of system CM}} \tag{3.5}$$

Using the TPR of the system and four error rates that have been evaluated above, we can find one of our objective function values, cost response by following formula.

$$Cost = C1 \times E_{CR}^{1} + C2 \times E_{CR}^{2} + C3 \times E_{CR}^{3} + C4 \times E_{CR}^{4} - C5 \times system\,TPR \tag{3.6}$$

The cost coefficient which the author employs is noise factor and it has low level of [1, 1, 1, 1, 2] and high level of [5, 1, 1, 1, 2]. The cost is evaluated by summing the sum of error costs and multiplication of system TPR its negative coefficient. This is because we need to consider two different effects such as the loss due to errors and the advantage owing to high TPR. If we look at two levels of cost coefficient, the only thing is being considered as a noise factor is the cost for fratricide. Rather than analyzing for all cost coefficients, this research focuses on the error that gives rise to fratricide.

The other response in this model is the sum of label accuracies, $P(E\,|\,"E") + P(F\,|\,"F") + P(C\,|\,"C")$. The TPR of the real system defined as $P("E"\,|\,E)$ can be one of post system (or operation) analysis to measure the performance and effectiveness of one executed operation, In contrast, a warfighter actually does not want the TPR of the system, $P("E"\,|\,E)$ but rather $P(E\,|\,"E")$; they want to know the label accuracy of the target of interest to avoid tragedies such as fratricide, civilian attack, and so on before they make decision and firing. However, if we only emphasize the label accuracy of enemy we may find a bad solution. We may find the parameter setting that maximizes the enemy label accuracy with poor friend label accuracy and clutter label accuracy.

The label accuracies, $P(E\,|\,"E")$, $P(F\,|\,"F")$ and $P(C\,|\,"C")$ can be calculated by the Bayes' theorem and the law of total probability [26] and it is shown as follows:

$$P(E\,|\,"E") = \frac{P("E"\,|\,E)P(E)}{P("E"\,|\,E)P(E) + P("E"\,|\,F)P(F) + P("E"\,|\,C)P(C)}$$

$$\approx \frac{\text{first row and column of system CM}}{\text{sum of first row of system CM}} \tag{3.7}$$

$$P(F\,|\,"F") = \frac{P("F"\,|\,F)P(F)}{P("F"\,|\,E)P(E) + P("F"\,|\,F)P(F) + P("F"\,|\,C)P(C)}$$

$$\approx \frac{\text{second row and column of system CM}}{\text{sum of second row of system CM}} \tag{3.8}$$

$$P(C\,|\,"C") = \frac{P("C"\,|\,C)P(C)}{P("C"\,|\,E)P(E) + P("C"\,|\,F)P(F) + P("C"\,|\,C)P(C)}$$

$$\approx \frac{\text{third row and column of system CM}}{\text{sum of third row of system CM}}. \tag{3.9}$$

In this paragraph, the two responses for this study are explained. This kind of procedure is performed at each of 160,000 design points. Thus, same number of costs and the differences of label accuracy are created and also the TPR of the system and each error rate come into being to be used as constraints.

Finding the Feasible Region

After getting the responses and other output values, we find the feasible region that satisfies the constraints. Before we determine the feasible region, we need to take an average of system responses for 10,000 different controllable factors (threshold combinations or Prior CM combinations). We obtain 16 cases of responses by employing four noise factors with two levels for one specific threshold pair (Dec (FPR, TPR), Class (FPR, TPR)). By taking an average, we can get average values in terms of E_{CR}^{i} (i = 1, 2, 3, 4) and the system *TPR* for 10,000 different controllable factors. Then we find the feasible region by comparing each average response with its critical value in the following equations.

$$E_{CR}^{i} \le \text{maximum Error rate}(i),\ i = 1, 2, 3, 4 \tag{3.10}$$

$$\text{system}\,TPR \ge \text{minimum}\,TPR \tag{3.11}$$

The maximum error rate and the minimum TPR of the system are affected by the quality of ROC curves. This is because if we use low quality ROC curves and high critical values, it is hard to find threshold combinations which satisfy constraints and thus, it is hard to construct feasible region

Finding Optimal Threshold Combinations

After constructing the feasible region, the optimal ROC threshold (prior CM) combinations are analyzed by LCDO, Taguchi's S|N ratio method and RPD with a combined array. When the objective function is the cost, in the case of LCDO, the threshold combination which creates the minimum average cost at the crossed array matrix is the solution, in the case of Taguchi's S|N ratio method the parameter which has minimum S|N ratio value is the answer and in the case of RPD, the response models of RPD and feasible region are used to investigate the optimal threshold combination.

Next, when the objective function is the sum of label accuracies, in the case of LCDO, we just try to find the prior CM combination that makes the minimum average value at the crossed array matrix, in the case of Taguchi's S|N ratio method, the prior CM combination which has the largest S|N ratio value is the solution and in the case of RPD with a combined array we make our judgment by overlapping feasible regions with its response models.

Evaluation of Simulation Output

The evaluation methods were briefly explained previously. In this part we consider again the meaning of two responses and we will formulate problem according to three evaluation methods: LCDO, Taguchi's S|N ratio method, and RPD using a combined array.

The sum of the label accuracies can be optimized by maximization. Why do we use maximization and what is the advantage of maximizing the sum? From the CM of the system, the diagonals represent correct evaluations and they are the sum proportional to label accuracies, $P(E\,|\,"E") + P(F\,|\,"F") + P(C\,|\,"C")$. Thus, we may get the solutions by maximizing sum of accuracies and this leads to minimize the off-diagonals components that usually make error costs. By the way, there is a tradeoff such as a joint increment of the FPR and the TPR at a ROC curve. Instead of finding the robust threshold combination we may lose the system's measure of effectiveness. Thus, if we use several constraints at the LCDO and Taguchi's S|N ratio method we can guarantee of minimum measure of effectiveness. However, in the case of RPD with a combined array, we may not use constraints if the feasible region does not match the region of robustness shown through RPD response model.

The other response is the cost and it has to be minimized as much as possible. Throughout the cost analysis we may gain understanding of the relationship between error rates and the TPR of the system. In other words, we may be able to find the relation between the costs due to mission accomplishment and the effectiveness of performing mission. For example, in the case of an offensive operation to destroy enemy, the commander seriously considers between the loss of his force by fratricide and a daring attack for increasing effectiveness (including the importance of time, location, mission and so on). We set the cost coefficient as a noise factor because, no one can predict the battlefield priority order and particularly, fratricide is given two different levels. We also expect to see whether the two response types have same solution or not.

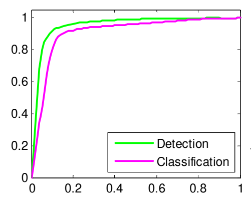
LCDO

<u>Goal: Determine Optimal CID system</u>

Decision factors:

1. The label accuracy
2. Costs of misdetection and that of misclassification are available
3. Constraints must be met
   - Acceptable Error rate (incorrect fire control decisions)
   - Acceptable minimum TPR

CID system and Evaluation (LCDO, Taguchi)



$$X = (x^D, x^C) = (\mathrm{Dec}(FPR, TPR),\ \mathrm{Class}(FPR, TPR))$$

CID Simulation

$$\mathrm{Response}\ [Cost, TPR, E_{CR}i (i = 1, 2, 3, 4)]$$

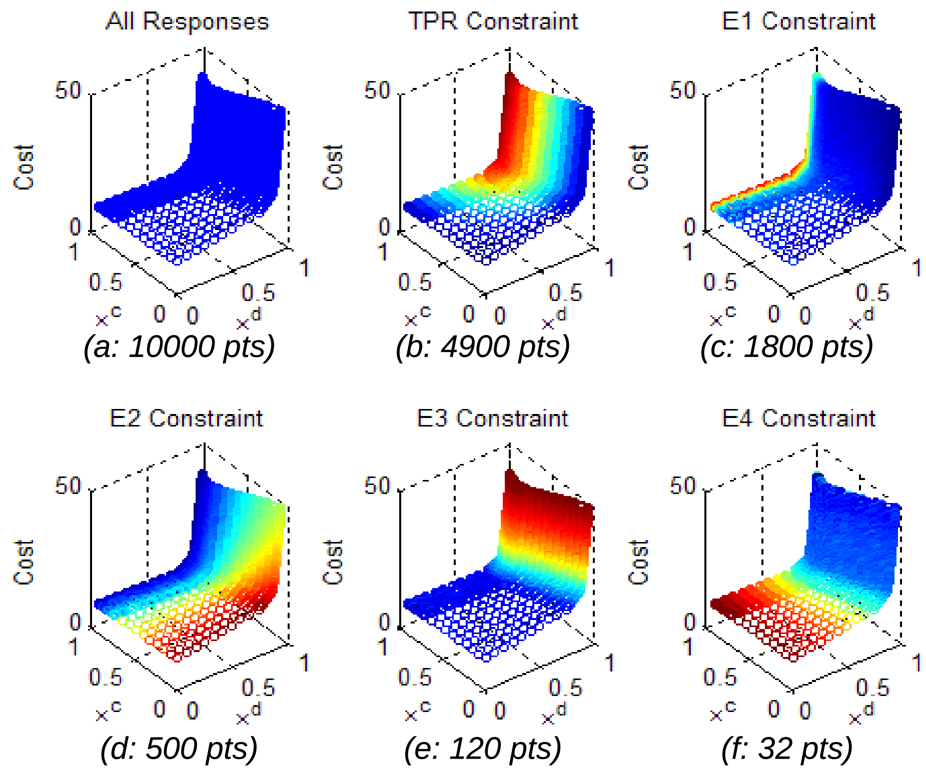


Figure 16: CID Optimization Goal and its Evaluation Example

Figure 16 represents the variable of this simulation and the procedure of evaluation by LDCO and Taguchi's S|N ratio method. For employing two methods, we need first the values of responses and those of constraints. After we found our responses, we try to shrink our threshold design points region by employing constraints. The example of this constraint is described at figure 16 in terms of the number of feasible points when the constraints are being accumulated. Except for the cost response graph (a), the other five represent the level of its constraint at each cost response by color index. Thus, the deep red of (b) at figure 16 is good performance since; the constraint is the TPR of the system and the deep blue reflects good performance at (c) ~ (f) of figure 16 because their color indices represent error rates.

Objective function

The sum of label accuracies approach:

$$\arg\max_{x\in X}\ Acc(x) = P_{(E|"E")}(x) + P_{(F|"F")}(x) + P_{(C|"C")}(x) \text{ maximize } Acc(x)$$

$$X = (x^D, x^C) = (\text{Dec}(FPR,TPR),\ \text{Class}(FPR,TPR))$$

The Cost approach:

$$\arg\min_{x\in X}\ Cost(x) = \sum_{i=1}^{4} C_i \cdot \text{E}^{i}_{\text{CR}}(x) - C_5 \cdot\ TPR(x) \ \text{ minimize } Cost(x)$$

$$X = (x^D, x^C) = (\text{Dec}(FPR,TPR),\ \text{Class}(FPR,TPR))$$

Subject to:

$E^1_{CR}(x) < \Pi_1$ limit incorrect fire decisions, fratricide

$E^2_{CR}(x) < \Pi_2$ limit missed classification of enemy as friend

$E^3_{CR}(x) < \Pi_3$ limit incorrect detection of clutter as enemy or friend

$E^4_{CR}(x) < \Pi_4$ limit missed detection of enemy or friend

$TPR(x) > \Pi_5$ guarantee minimum system TPR, the true positive rate

Taguchi's S|N Ratio Method

We can apply same kind of procedure after getting Taguchi's S|N ratios from the crossed array design. The original formula of S|N_S (signal to noise ratio smaller better) has been modified a little bit by eliminating its negative coefficient since the cost response is not less than 1.0.

Objective function:

The sum of label accuracies approach:

$$\arg\max_{x \in X} \ \text{S|N}_\text{L}(x) = -10\log\left(\frac{1}{n}\sum_{i=1}^{n}\frac{1}{y_i^2}\right), \quad n = \text{the \# of outer array combination}$$

$$X = (x^D, x^C) = (\text{Dec}(FPR, TPR),\ \text{Class}(FPR, TPR))$$

$y_i = accuracy$ response in the crossed array,

maximize $\text{S|N}_\text{L}(x)$, the signal to noise ratio

The Cost approach:

$$\arg\min_{x \in X} \ \text{S|N}_\text{S}(x) = 10\log\left(\frac{1}{n}\sum_{i=1}^{n} z_i^2\right),$$

n = the # of outer array combiniation,

$$X = (x^D, x^C) = (\text{Dec}(FPR, TPR),\ \text{Class}(FPR, TPR))$$

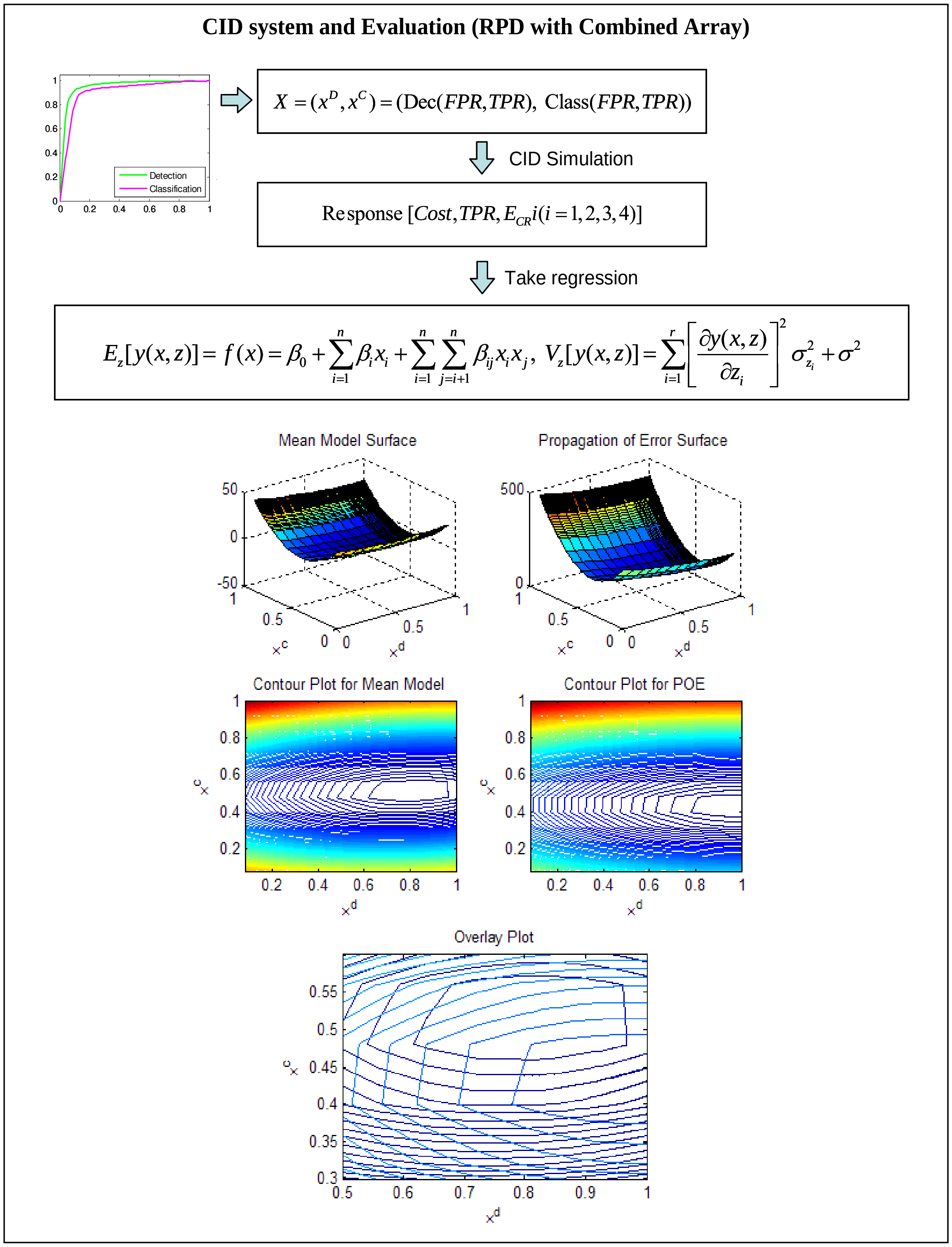
$z_i = Cost$ response in the crossed array,

minimize $\text{S|N}_\text{S}(x)$, the signal to noise ratio

Subject to:

Same as those of LCDO for both approaches

RPD with Combined Array Design



CID system and Evaluation (RPD with Combined Array)
Detection
Classification
$X = (x^D, x^C) = (\mathrm{Dec}(FPR, TPR),\ \mathrm{Class}(FPR, TPR))$
CID Simulation
Response $[Cost, TPR, E_{CR} i (i = 1, 2, 3, 4)]$
Take regression
$E_z[y(x,z)] = f(x) = \beta_0 + \sum_{i=1}^{n} \beta_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \beta_{ij} x_i x_j,\ V_z[y(x,z)] = \sum_{i=1}^{r} \left[\frac{\partial y(x,z)}{\partial z_i} \right]^2 \sigma_{z_i}^2 + \sigma^2$
Mean Model Surface
Propagation of Error Surface
Contour Plot for Mean Model
Contour Plot for POE
Overlay Plot
x^c
x^d

Figure 17: CID Evaluation Example at RPD

Figure 17 shows the procedure of evaluation by RPD with a combined array design matrix. After finishing the simulation for all threshold combinations, we first do a regression and make a mean model and a propagation of error model. Then the contour plots for those models are constructed and overlapping figure is also made. By comparing the value of the mean and the propagation error we can find subjective robust point(s).

Objective function:

$$X = (x^D, x^C) = (\mathrm{Dec}(FPR, TPR),\ \mathrm{Class}(FPR, TPR))$$

Obtaining reliable threshold combinations rather than using a specific objective function

Subject to:

Possible use of LCDO constraints

IV. Experiments and Results

Introduction

This research employs two ROC curve sets for experiments: (1) one normal ROC curve for classification and a little better one for detection, (2) much improved ROC curves for both but still the detection curve is better than the classification's curve. We can see how the ROC curves look like at the beginning of every experiment. As we know through the previous Chapters, we have two responses and three evaluation methods. At every ROC curve set, we will get three (if we find one answer at RPD with a combined array design approach) or more solutions (threshold combinations) for each response. The critical values for error rates and the TPR have not been applied in these experiments thus the feasible region is all design points, all possible threshold combinations. In addition, the confirmation experiments are carried out at the end of the experiments for each ROC curve set.

1st ROC curve set

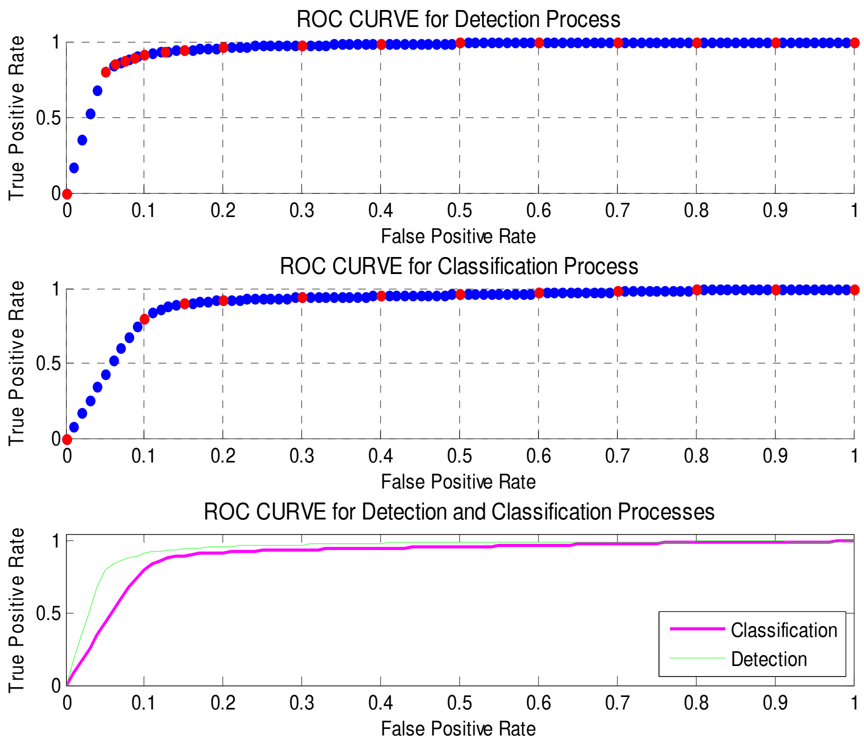


Figure 18: ROC Curves for 1st Experiment Set

These ROC curves are created by RBFs [16]. The red points at the first two graphs have been utilized to erect two ROC curves. From the ROC curves, we gather one hundred pairs of ((FPR), (TPR)) for detection and classification and thus, the total number of ROC threshold combinations is 10,000.

The Cost Response Approach

For the solutions of LCDO and Taguchi's S|N method, the crossed array design matrix is required since we need the average cost and the $S|N_S$ across outer array.

Table 7: The Solutions of the Cost Response of the 1st ROC Set

Method	Comb #	TPR_D	TPR_C	FPR_D	FPR_C	Cost	S|N S
LCDO	1704	0.6816	0.9153	0.04	0.18	39.8311	33.0715
SN_L	1704	0.6816	0.9153	0.04	0.18	39.8311	33.0715
RPD	803	0.524	0.7435	0.03	0.09	45.7075	34.1725

The solution of the LCDO and that of the Taguchi method are same since the smallest value of $S|N_S$ and that of the cost are occurred in the same threshold combination. The answer of the RPD with a combined array is the 803rd combination and it is developed by following procedure.

Table 8: The Regression Analysis results of RPD Using the Cost Response of the 1st ROC Set

		DoF	MS	F	P-value
SS reg	10558641032	17	621096531.3	26257.5058	0
SS res	3784223296	159982	23654.0567		
SSt	14342864329	159999			
R^2	0.7362				

	Coefficient	Std Error	T	T-crit	P-value
Intercept	-194.9538	3.4819	-55.9899	1.96	0
X1	63.8489	3.3826	18.8755	1.96	0
X2	-19.2512	3.5119	-5.4817	1.96	0
X1^2	475.2185	2.544	186.8023	1.96	0
X2^2	18.6246	2.1806	8.5409	1.96	0
X1X2	-8.8243	3.6481	-2.4189	1.96	0.0156
Z1	25.0018	1.5163	16.4884	1.96	0
Z2	27.6157	1.5163	18.2122	1.96	0
Z3	5.018	1.5163	3.3093	1.96	0.0009
Z4	18.2774	1.5163	12.0537	1.96	0
X1Z1	226.616	1.3492	167.9617	1.96	0
X1Z2	-14.1357	1.3492	-10.477	1.96	0
X1Z3	-12.4069	1.3492	-9.1957	1.96	0
X1Z4	1.6665	1.3492	1.2351	1.96	0.2168
X2Z1	0.0032	1.0396	0.0031	1.96	0.9975
X2Z2	-23.0579	1.0396	-22.1792	1.96	0
X2Z3	7.9989	1.0396	7.6941	1.96	0
X2Z4	-19.7686	1.0396	-19.0153	1.96	0

Variable Definition

X1: TPR_D, X2: TPR_C, Z1: Map size, Z2: # of Enemy, Z3: # of Friend, Z4: Cost coefficient

$$y_{\text{cost}} = f(x) + h(x,z) + \varepsilon \tag{3.1}$$

$$f(x) = -194 + 63.85x_1 - 19.25x_2 + 475.22x_1^2 + 18.62x_2^2 - 8.82x_1x_2 \tag{3.2}$$

$$\begin{aligned} h(x,z) = {} & 25z_1 + 27.62z_2 + 5.02z_3 + 18.28z_4 + 226.62x_1z_1 - 14.14x_1z_2 - 12.41x_1z_3 \\ & + 1.67x_1z_4 + 0x_2z_1 - 23.06x_2z_2 + 8x_2z_3 - 19.77x_2z_4 \end{aligned} \tag{3.3}$$

As shown by the regression result, all but X1Z4 and X2Z1 coefficients are statistically different than zero. The reason there is no interaction between the TPR of the detection (X1) and the cost coefficient (Z4) might be explainable since our cost coefficient, one of noise factors describes only the fratricide cost and it is primarily determined by classification process. The map size (Z1) represents the size of a ROI and actually it reflects the clutter which is usually identified through the detection process (X1), not the classification process (X2). The model also has a reasonable R-squared value of .7362. To get the best answer in RPD with a combined array, this model is then used to develop surfaces for the response mean and propagation of error as shown below [22]. These plots are then combined into a contour plot to estimate the robust parameters.

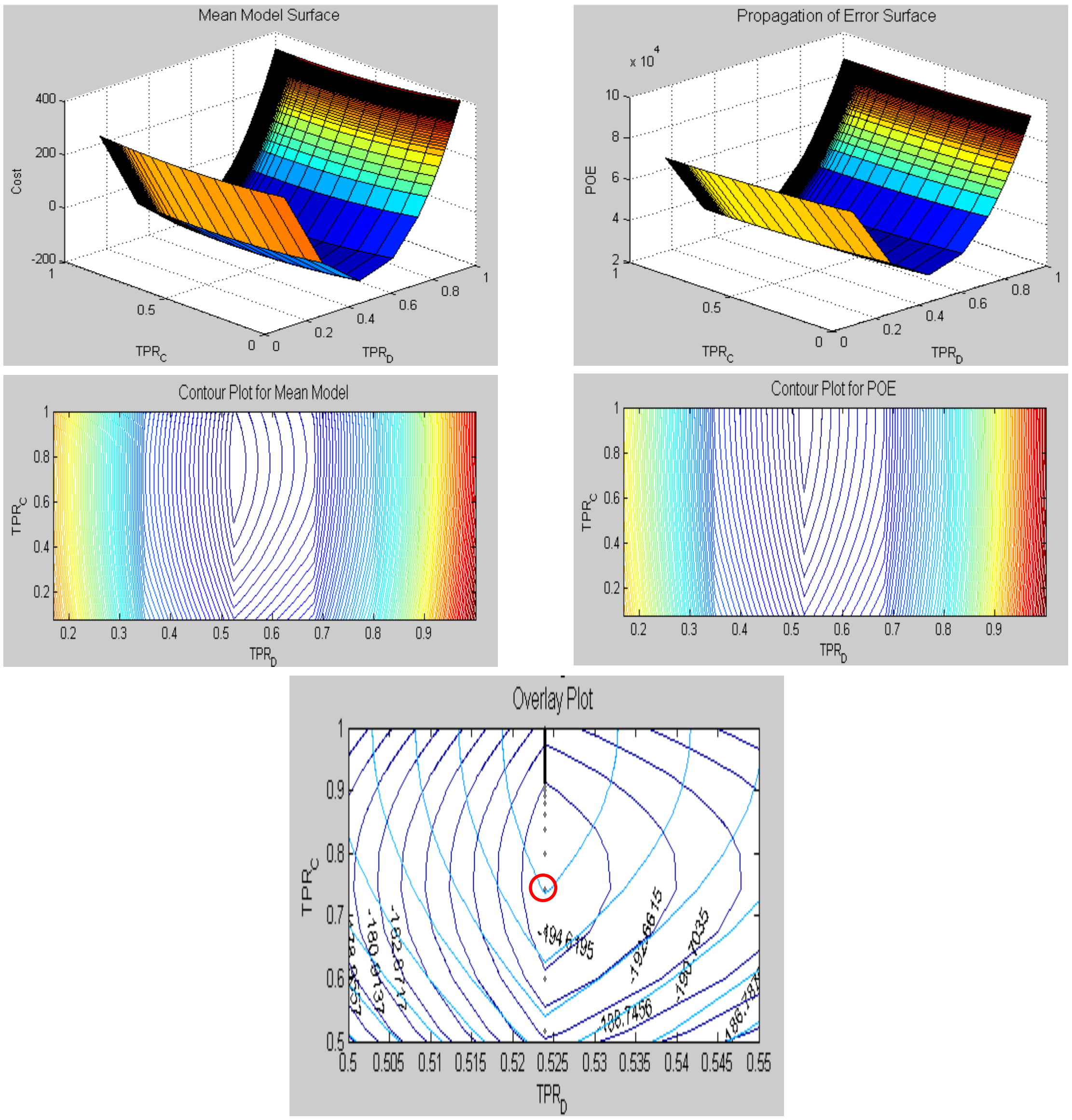


Figure 19: Surface, Contour and Overlay Plots for RPD Using the Cost Response of the 1st ROC Set

The above overlay plot shows that the minimal error-cost takes place at the red circle that is the point of the 803rd combination. Due to tradeoff between the FPR and the TPR, the robust solution might appear around the middle of the design space. At the confirmation experiment each solution found by each technique will be investigated to prove which one works well in different cases.

The Label Accuracy Response Approach

In this section, the experiment is carried out with respect to the sum of label accuracies and the same methods of finding solutions are employed.

Table 9: The Solutions the Label Accuracy Response of the 1st ROC Set

Method	Comb #	TPR_D	TPR_C	FPR_D	FPR_C	Sum of Acc	S|N L
LCDO	1306	0.8443	0.8919	0.06	0.14	0.8904	38.9399
SN_L	1306	0.8443	0.8919	0.06	0.14	0.8904	38.9399
RPD 1st	904	0.6816	0.8	0.04	0.1	0.865802	38.5304
RPD 2nd	1005	0.8	0.8375	0.05	0.11	0.885893	38.8577

The solutions of the LCDO and the Taguchi's method are the same since the largest value of S|N ratio and that of accuracy occupied same threshold combination. The two subjective solutions of RPD with a combined array have been collected by following way.

Table 10: The Regression Analysis Results of RPD Using the Label Accuracy Response of the 1st ROC Set

		DoF	MS	F	P-value
SS reg	4093.6423	17	240.8025	7500.9201	0
SS res	5135.9117	159982	0.0321		
SSt	9229.554	159999			
R^2	0.4435				

	Coefficient	Std Error	T	T-crit	P-value
Intercept	1.2385	0.0041	305.3164	1.96	0
X1	-0.0297	0.0039	-7.5256	1.96	0
X2	0.0366	0.0041	8.9432	1.96	0
X1^2	-0.7081	0.003	-238.9415	1.96	0
X2^2	-0.1025	0.0025	-40.3634	1.96	0
X1X2	-0.001	0.0042	-0.234	1.96	0.815
Z1	0.1066	0.0018	60.3436	1.96	0
Z2	-0.0738	0.0018	-41.7539	1.96	0
Z3	-0.0276	0.0018	-15.6097	1.96	0
Z4	0	0.0018	0	1.96	1
X1Z1	-0.1611	0.0016	-102.5249	1.96	0
X1Z2	0.0849	0.0016	54.0217	1.96	0
X1Z3	0.0683	0.0016	43.4402	1.96	0
X1Z4	0	0.0016	0	1.96	1
X2Z1	0.0019	0.0012	1.5436	1.96	0.1227
X2Z2	0.0423	0.0012	34.9091	1.96	0
X2Z3	-0.0441	0.0012	-36.3765	1.96	0
X2Z4	0	0.0012	0	1.96	1

Variable Definition

X1: TPR_D, X2: TPR_C, Z1: Map size, Z2: # of Enemy, Z3: # of Friend, Z4: Cost coefficient

$$y_{\text{acc}} = f(x) + h(x,z) + \varepsilon \tag{3.4}$$

$$f(x) = 1.2385 - 0.0297x_1 + 0.0366x_2 - 0.7081x_1^2 - 0.1025x_2^2 - 0.001x_1x_2 \tag{3.5}$$

$$\begin{aligned} h(x,z) = {} & 0.1066z_1 - 0.0738z_2 - 0.0276z_3 + 0z_4 - 0.1611x_1z_1 + 0.0849x_1z_2 \\ & - 0.0683x_1z_3 + 0x_1z_4 + 0.0019x_2z_1 + 0.0423x_2z_2 - 0.0441x_2z_3 - 0x_2z_4 \end{aligned} \tag{3.6}$$

As shown above, the X1X2, X2Z1, Z4 and its interactions with X1 and X2 are not statistically different than zero. The assumption of the independence between detection and classification system might be the reason of no interaction between X1 and X2. And the explanation of no interactive effect between X2 and Z1 might be same as the case of RPD in the cost approach. Z4 is the factor of the cost coefficient and it does not work in the accuracy response approach due to no cost evaluation. The model has the low R-squared value of .4435 and this means the model does not describe a large portion of the variance in the response.

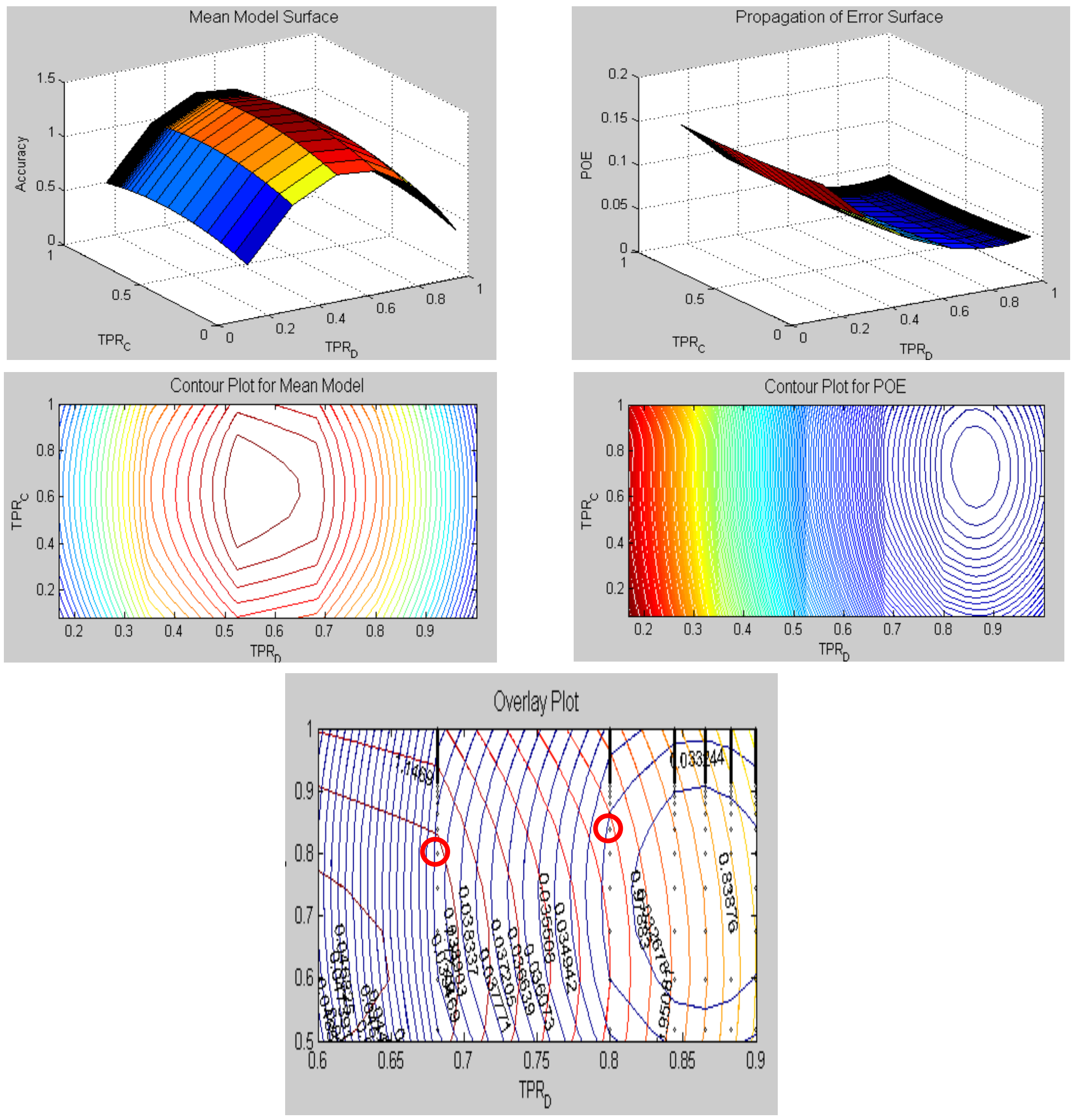
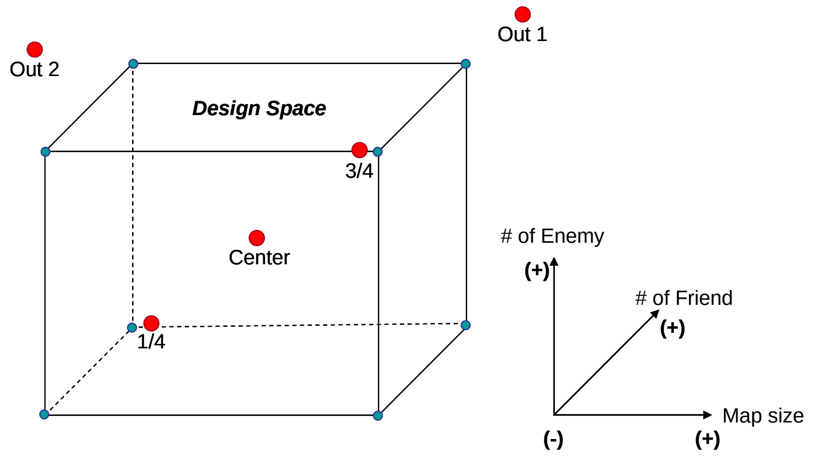


Figure 20: Surface, Contour and Overlay Plots for RPD Using the Label Accuracy Response of the 1st ROC Set

As shown by figure 20, the higher total accuracy roughly happens at the middle of the mean model and the lower POE turns out at the northeast quadrat of the POE model. Due to difficulty in interpretation of the POE function and low R-squared value, the 904th and 1005th design points displayed on the overlay plot are selected as the solutions of RPD. The 904th point has higher response and POE than the 1005th point. We can compare which one shows better performance at the confirmation experiment that is, better result at the 904th means that mean model dominated selection is needed when we figure out overlay plot and vice versa.

Confirmation Experiment for 1st ROC set



Test point	Map size	# of Enemy	# of Friend	Coefficient
Center	450	18	18	[3 1 1 1 2]
1/4	225	9	9	[2 1 1 1 2]
3/4	675	27	27	[4 1 1 1 2]
Out1	9000	360	360	[40 1 1 1 2]
Out2	2000	10	70	[10 1 1 1 2]

Figure 21: Notional Example of Design Space and the Table of Confirmation Experiments

The confirmation experiment with regards to the 1st ROC set is conducted in different ROI surroundings: (1) the center point of the design, (2) one fourth point of the design, (3) three fourth point of the design and (4) two points out of the design. The confirmation experiment for both responses is performed together and the values for each response are also reported together.

Table 11: The Confirmation Experiments Results of the 1st ROC Set

Response Type	Method Type	Comb #	Cost					Ave_acc
			Center	1/4	3/4	Out1	Out2	
min Cost	**LCDO & Taguchi**	**1704**	32.04	14.95	50.28	1433.31	115.15	0.931
	RPD	**803**	37.03	16.83	59.56	2543.79	111.88	0.928
max Acc	**LCDO & Taguchi**	**1306**	35.98	16.57	57.48	1962.89	143.57	0.924
	RPD1	**904**	35.56	16.01	57.51	2539.57	119.51	0.930
	RPD2	**1005**	35.13	15.61	56.70	2457.91	129.76	0.930
mean			222.24	109.49	336.54	5615.20	1013.80	0.507
mean of the best 10%			41.63	19.76	64.20	1196.70	159.03	0.912

The 1704th combination generally shows good result and it also has the maximum accuracy. The values shaded blue show the best performance when we do the confirmation experiment with the solutions of each evaluation method for a given design point. We can also see how the results are close to the mean and optimal values by looking at the last two rows of table 11. In most cases, the values of blue are better than the mean of the best 10% and actually, they are close to the mean of the best 5%. If we think about the accuracy, even though the threshold combination that had the maximum accuracy in the design points, it does not attain the biggest at the confirmation experiments. But, rather, the point that has the smallest cost has the biggest accuracy at table 11. In addition, it is hard to say that there is a difference between two solutions of RPD with a combined array of the accuracy response.

2nd ROC curve set

The ROC curves for the CID system are generally determined by the quality of signals and the selection of the decision threshold [27]. If the 1st set of ROC curves has a low quality of signal and hence the region of intersection between the target probability distribution and the clutter probability distribution in the case of detector is relatively large, the 2nd ROC curve set comes up with high quality of signals. Thus, we can expect improved ROC curve behaviors and those are demonstrated at figure 22.

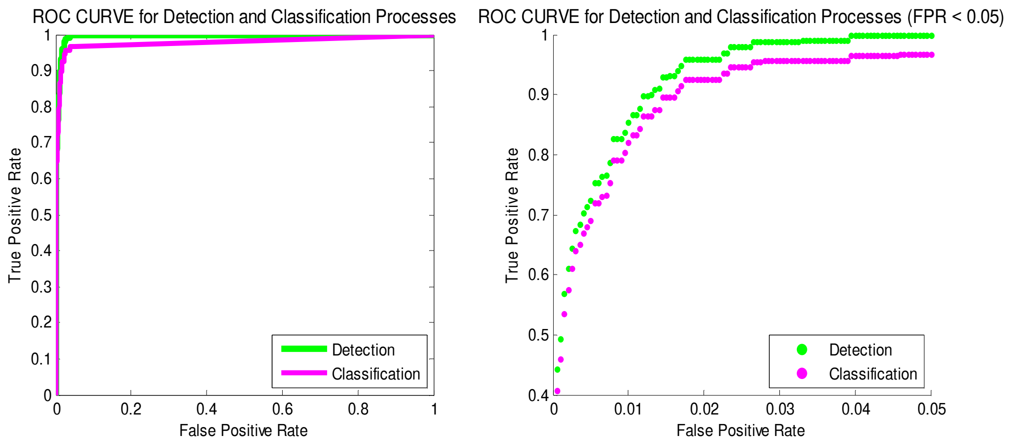


Figure 22: ROC Curves for 2nd Experiment Set

As you see, the ROC curves for 2nd set are much better than previous ones in terms of their high TPR at the same FPR. Right-hand side graph of figure 22 is used for this experiment and its range of x-axis (FPR) is (0, .05) for both curves. Due to different ROC curves we may see very different results as compared with the 1st ROC set.

The Cost Response Approach

Table 12: The solutions of the cost response of the 2nd ROC set

Method	Comb #	TPR_D	TPR_C	FPR_D	FPR_C	Cost	S|N S
LCDO	7930	0.9307	0.9661	0.015	0.04	11.4907	22.7413
SN_L	7924	0.8981	0.9661	0.012	0.04	11.5961	22.5422
RPD	9816	0.8261	0.9667	0.008	0.0495	12.9393	23.2914

In contrast with 1st ROC set of the cost response approach, three different solutions are estimated but there is still no big difference between the solution of LCDO and that of Taguchi method. Again, the following tables and figures explain how we get the solution of RPD with a combined array.

Table 13: The Regression Analysis Results of RPD Using the Cost Response of the 2nd ROC Set

		DoF	MS	F	P-value
SS reg	56347668.2	17	3314568.7	58921.006	0
SS res	8999699.296	159982	56.2544		
SSt	65347367.49	159999			
R^2	0.8623				

	Coefficient	Std Error	T	T-crit	P-value
Intercept	26.0934	0.0784	332.7948	1.96	0
X1	-1.0333	0.0974	-10.6117	1.96	0
X2	-13.8853	0.0974	-142.5714	1.96	0
X1^2	13.0009	0.0811	160.4053	1.96	0
X2^2	0.5499	0.081	6.7892	1.96	0
X1X2	-5.241	0.1022	-51.2999	1.96	0
Z1	2.737	0.0482	56.7722	1.96	0
Z2	18.7106	0.0482	388.1022	1.96	0
Z3	4.7785	0.0482	99.1165	1.96	0
Z4	11.9501	0.0482	247.8725	1.96	0
X1Z1	12.0206	0.0438	274.546	1.96	0
X1Z2	-3.7072	0.0438	-84.6722	1.96	0
X1Z3	-5.2242	0.0438	-119.3194	1.96	0
X1Z4	1.4096	0.0438	32.1948	1.96	0
X2Z1	-0.0023	0.0438	-0.0531	1.96	0.9577
X2Z2	-13.5045	0.0438	-308.6534	1.96	0
X2Z3	0.4343	0.0438	9.9273	1.96	0
X2Z4	-11.5835	0.0438	-264.7483	1.96	0

Variable Definition

X1: TPR_D, X2: TPR_C, Z1: Map size, Z2: # of Enemy, Z3: # of Friend, Z4: Cost coefficient

$$f(x) = 26.09 - 1.03x_1 - 13.89x_2 - 13x_1^2 - 0.55x_2^2 - 5.24x_1x_2 \tag{3.8}$$

$$\begin{aligned} h(x,z) = {} & 2.746z_1 + 18.71z_2 + 4.78z_3 + 11.95z_4 - 12.02x_1z_1 - 3.71x_1z_2 \\ & - 5.22x_1z_3 + 1.41x_1z_4 + 0x_2z_1 - 13.5x_2z_2 + 0.43x_2z_3 - 11.58x_2z_4 \end{aligned} \tag{3.9}$$

As shown above regression results, the interaction between the TPR of the classification (X2) and the map size (Z1) is the only one that does not influence to the model. The R-squared value of .8623 is acceptable.

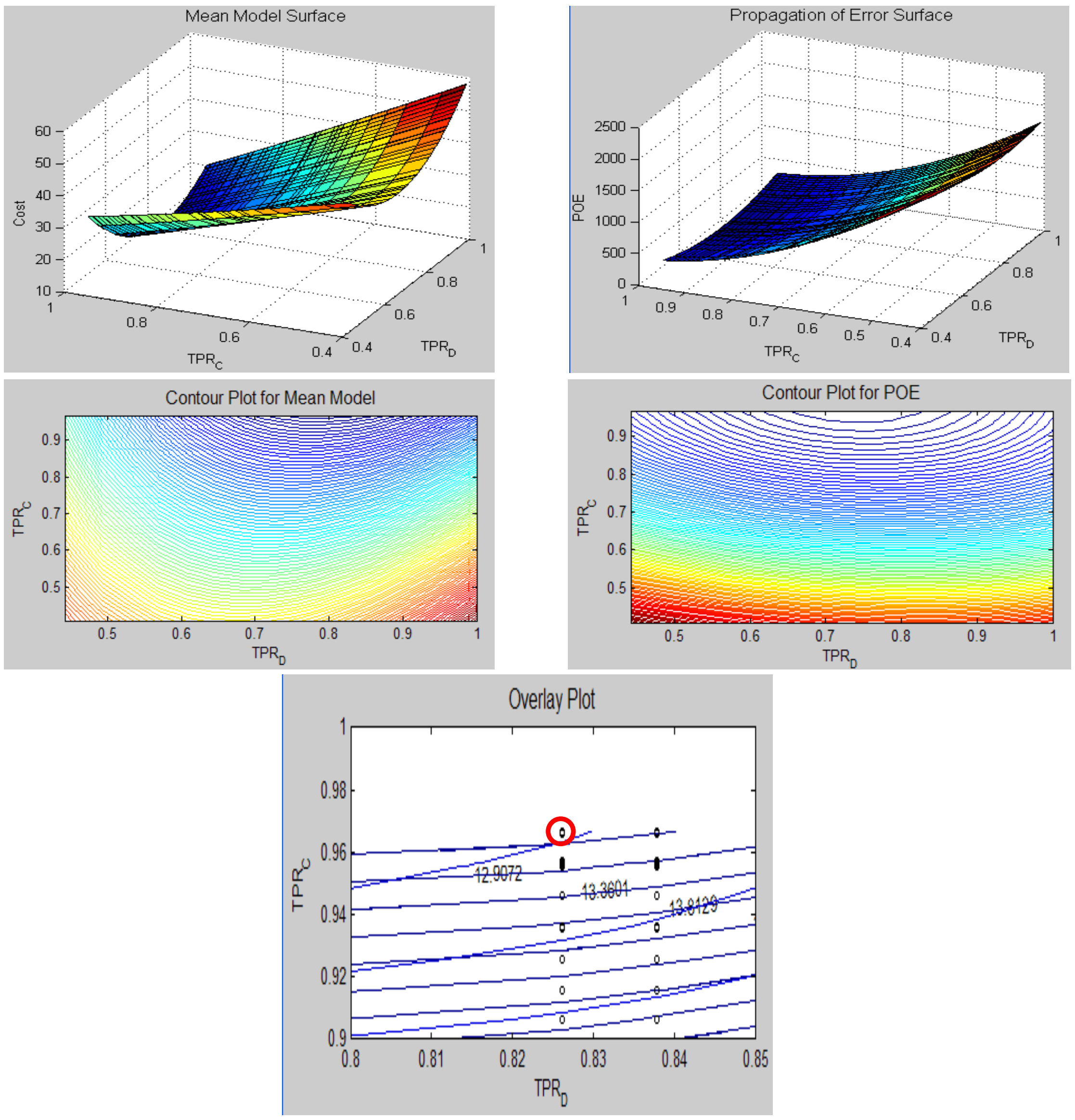


Figure 23: Surface, Contour and Overlay Plots for RPD Using the Cost Response of the 2nd ROC Set

Again, when the maximum values of FPR for both are .05, the TPR of the detection is 1 and that of the classification is .967. The solution of the RPD with a combined array method is appeared at overlay plot with lower mean and POE value and it is the 9816th combination.

The Label Accuracy Response Approach

Table 14: The Solutions of the Label Accuracy Response of the 2nd ROC Set

Method	Comb #	TPR_D	TPR_C	FPR_D	FPR_C	Sum of Acc	S|N L
LCDO	5253	0.989	0.9554	0.0265	0.0265	0.9686	39.7228
SN_L	5253	0.989	0.9554	0.0265	0.0265	0.9686	39.7228
RPD1	9347	0.9795	0.9667	0.0235	0.047	0.9675	39.7117
RPD2	9992	1	0.9667	0.046	0.05	0.9551	39.6007

We get same solution, 5253rd combination for LCDO and Taguchi method, and two slightly different answers for RPD with a combined array. However, the location between solutions is not close like those of 1st ROC set.

Table 15: The Regression Analysis Results of RPD Using the Label Accuracy Response of the 2nd ROC Set

		DoF	MS	F	P-value
SS reg	397.9119	17	23.4066	47894.4	0
SS res	78.1852	159982	0.0005		
SSt	476.0971	159999			
R^2	0.8358				

	Coefficient	Std Error	T	T-crit	P-value
Intercept	0.9043	0.0002	3912.9892	1.96	0
X1	0.0491	0.0003	171.1542	1.96	0
X2	0.0242	0.0003	84.4045	1.96	0
X1^2	-0.0194	0.0002	-81.0572	1.96	0
X2^2	-0.0029	0.0002	-12.278	1.96	0
X1X2	0.0087	0.0003	28.944	1.96	0
Z1	0.0822	0.0001	578.3533	1.96	0
Z2	-0.0518	0.0001	-364.6208	1.96	0
Z3	-0.0263	0.0001	-185.1719	1.96	0
Z4	0	0.0001	0	1.96	1
X1Z1	-0.058	0.0001	-449.4661	1.96	0
X1Z2	0.0264	0.0001	204.5851	1.96	0
X1Z3	0.0287	0.0001	222.6852	1.96	0
X1Z4	0	0.0001	0	1.96	1
X2Z1	-0.0236	0.0001	-182.6671	1.96	0
X2Z2	0.0248	0.0001	192.0206	1.96	0
X2Z3	-0.0024	0.0001	-18.3741	1.96	0
X2Z4	0	0.0001	0	1.96	1

Variable Definition

X1: TPR_D, X2: TPR_C, Z1: Map size, Z2: # of Enemy, Z3: # of Friend, Z4: Cost coefficient

$$f(x) = 0.9043 + 0.0491x_1 + 0.0242x_2 - 0.0194x_1^2 - 0.0029x_2^2 - 0.0087x_1x_2 \qquad (3.9)$$

$$\begin{aligned} h(x,z) = 0.0822z_1 - 0.0518z_2 - 0.0263z_3 + 0z_4 - 0.058x_1z_1 + 0.0264x_1z_2 \\ 0.0287x_1z_3 + 0x_1z_4 - 0.0236x_2z_1 + 0.0248x_2z_2 - 0.0024x_2z_3 + 0x_2z_4 \end{aligned} \qquad (3.10)$$

The regression result for this experiment illustrates that the cost coefficient factor (Z4) and its interactions with controllable factors are redundant since the response is the accuracy. And the R-squared value of .8358 is suitable.

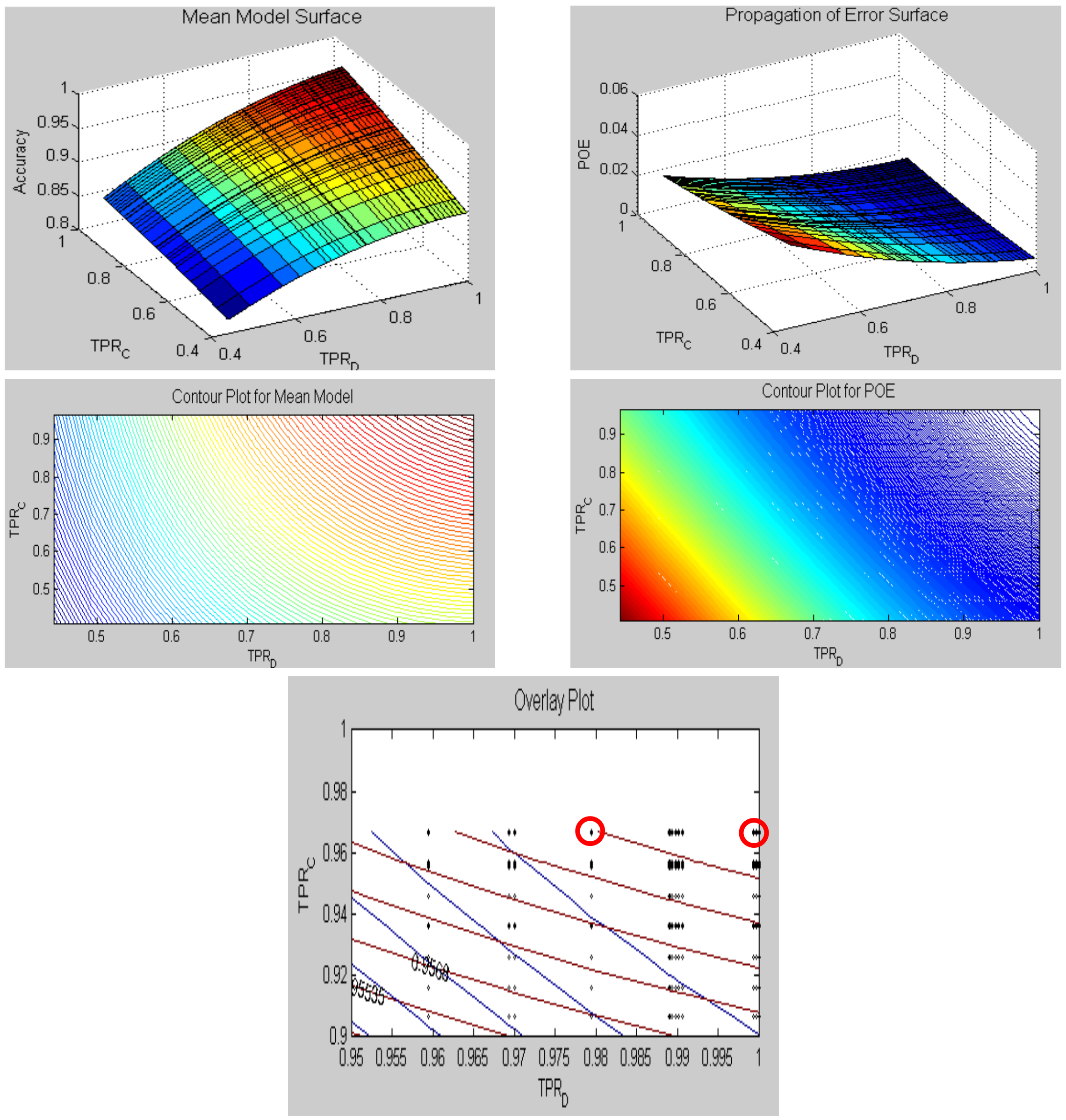


Figure 24: Surface, Contour and Overlay Plots for RPD Using the Label Accuracy Response of the 2nd ROC Set

By comparing mean and POE plots, we catch the points having high accuracy and low POE. However, the mean model possibly does not represent real behavior of controllable factors because, the regression tried to give us good fit employing all repressors. The following figure constructed from the crossed array design with mean accuracy as its response and it shows the difference that the maximum accuracy does not happen at the exact top of both axes (controllable factors) of the design space.

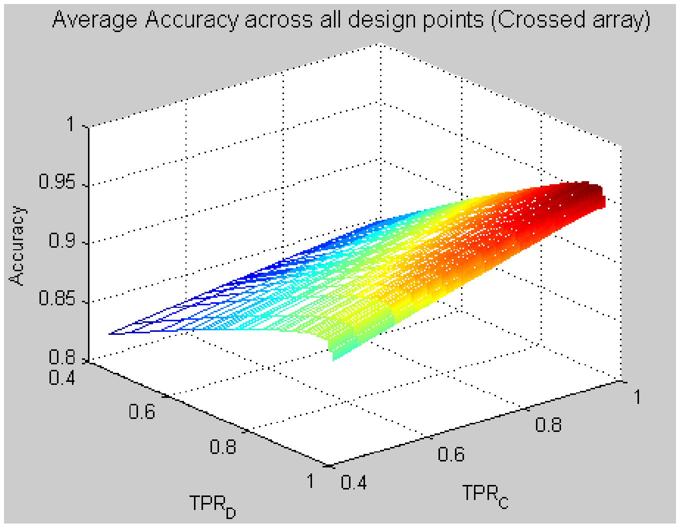


Figure 25: Average Accuracy across All Design Points from the Crossed Array

Confirmation experiment for 2nd set

Table 16: The Confirmation Experiments Results of the 2nd ROC Set

Response Type	Method Type	Comb #	Cost					Ave_acc
			Center	1/4	3/4	Out1	Out2	
min cost	**LCDO**	**7930**	9.28	3.57	15.68	638.04	38.56	0.979
	Taguchi	**7924**	9.26	3.50	15.65	621.70	35.24	0.979
	RPD	**9816**	10.27	4.08	16.85	602.55	33.42	0.978
max Acc	**LCDO & Taguchi**	5253	12.24	4.86	20.54	868.87	56.27	0.972
	RPD1	**9347**	11.18	4.37	18.68	696.44	51.44	0.974
	RPD2	**9992**	19.95	8.76	31.66	871.42	93.21	0.954
mean			17.66	7.08	30.10	1762.18	65.12	0.967
mean of the best 10%			9.75	3.64	16.46	628.70	34.73	0.978

The Confirmation experiment for this set is performed on the points of figure 21 and the results are presented at table 16. The 7924th combination shows good results in the design space and the 9816th combination operates well out of design space. Again, the values of blue display the best performance for a given design point. The last two rows of table 16 helps to understand how the results are close to the mean and optimal values. In all cases, the values shaded blue are better than the mean of the best 10%. Like the 1st ROC set, the biggest accuracy occurs around the points that make the smallest costs. There is something we need to know in this research's confirmation experiments. We made one table for the result but, in the case of cost response, we may focus on the cost values and just refer accuracy value in the case of accuracy response, it may be vice versa.

Summary of experiments results

In this Chapter, the experiments were carried out using two different ROC curve sets with the CID model and three output analysis techniques as explained in the previous Chapter. Further, the confirmation experiments were accomplished at the optimal parameters obtained from three techniques for each response. The summary of experiments and results are follows:

- In the case of the cost response, the solutions from LCDO and the Taguchi method worked well within the design space in terms of the performance of the confirmation experiments. The solution from RPD with a combined array gave us lower cost and higher accuracy outside of the design space.

- In the case of the accuracy response, the performances were slightly enhanced due to the solution of the RPD with a combined array in both sets.
- The parameters that achieved the minimum cost were very close to the parameters that yielded the maximum accuracy.
- The accuracy approach is more sensitive than the cost approach with respect to the design space. The solutions for the accuracy response did not show the largest average accuracy in the last column of table 16. On the other hand, the solutions from the cost approach showed the largest accuracy along with the smallest cost.

These results show that the parameters that yield the minimum cost also provide maximum accuracy. The cost response employed in this study is mainly comprised of four critical error rates (even though it has one more component, the TPR of the system) located in the off-diagonals of the CM; however, the accuracy response is the sum of the proportion of the diagonals. Thus, if we try to minimize the cost it automatically maximizes accuracy and vice versa. Throughout the results of confirmation experiments we can see this tendency.

V. Summary and Conclusions

Many studies related to CID have the same goal: to maximize combat/mission effectiveness while reducing total casualties due to enemy action and fratricide [5:1]. The objectives of this thesis were: (1) the modeling of a CID situation and (2) obtaining robust and controllable input variable settings. Considering the features of CID, input variables were defined as controllable and uncontrollable and the confusion matrices that are used in ROC theory were adapted to controllable factors.

For CID modeling this research employed the following assumptions: (1) each detector and classifier occupies a predetermined ROC curve, (2) a neutral force and civilian are in the clutter, (3) there are three characteristics in a virtual ROI such as: enemy object, a friendly object, and clutter, (4) all entities have to be declared one of these and no entity can be non-declared.

All results of the CID system are summarized by a posterior CM. Throughout the posterior CM analysis, the responses that the simulator wants to gather can be obtained. This study has two responses, cost and accuracy. The cost is evaluated by multiplying error rates of the CM and their cost coefficients and the accuracy is calculated by summing types of accuracies in the CM with respect to enemy, friend, and clutter.

To find optimal parameters for each response, three evaluation techniques were applied: (1) LCDO, (2) Taguchi's S|N ratio method, and (3) RPD with a combined array. For (1) and (2), the crossed array design that has controllable factors as its inner array and uncontrollable factors as its outer array was employed and for (3), obviously, the combined array design was used. The solutions of (1) and (2) are almost the same but that of (3) is different in two experiment sets.

We used confirmation experiments to compare the performance of each solution and the results were: (1) LCDO and the Taguchi method give us better output in the cost response and RPD with a combined array shows a slightly better performance in accuracy response, (2) in general LCDO and the Taguchi method can be applied within the design space and RPD with a combined array can be operated out of the design space, (3) the parameters that make the minimum cost yield the near maximum accuracy and (4) the accuracy approach is more sensitive than the cost approach with regards to the design space.

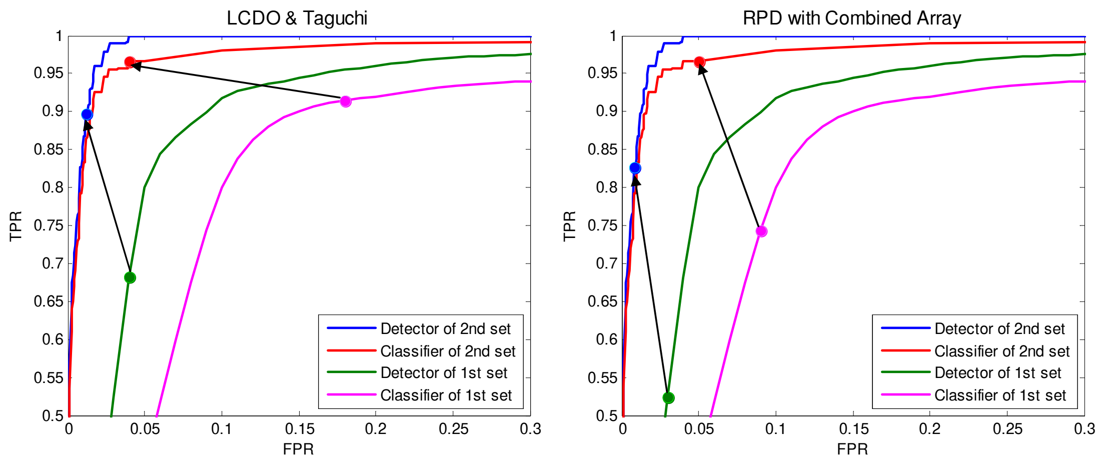


Figure 26: The Movement of the Optimal Points for Each Technique between Two ROC Sets

The optimal points for both detector and classifier in figure 26 moved to the points that allow higher TPR with lower FPR (north-west corner); particularly, the TPR of the detector for all evaluation methods increased significantly between the two ROC sets. This may represent the importance of finding a small number of objects in a huge ROI. Comparing two graphs in figure 26, the RPD with a combined array technique generally has a strong tendency of keeping lower FPR. Thus we can expect relatively lower error cost when we employ the RPD method since the error cost is primarily produced by a FPR, i.e. the FPR that triggers a fratricide or incorrect detection of a clutter.

Table 17: The Best Performance Values of the Confirmation Experiments for Each ROC Set

ROC Set Type	**Cost**					**Ave_acc**
	Center	**1/4**	**3/4**	**Out1**	**Out2**	
1st set	32.04	14.95	50.28	1433.31	111.88	0.931
2nd set	9.26	3.50	15.65	602.55	33.42	0.979

Table 17 is a table that reorganizes the best values of the confirmation experiments for each ROC set. As shown, the improvement of ROC curve behaviors in figure 26 induces a large decrease in the cost and a small increase in the accuracy. Even though the accuracy only increased 5% between the two ROC sets, the cost decreased more than 3.3 times. This implies a need for improvement in the ROC curves, the performance of the detector and classifier, to increase accuracy and decrease error cost.

In conclusion, if we consider the diverse characteristics of CID, the simulator needs to focus on finding the parameters that yield the maximum accuracy value. This is because minimum cost is accomplished at the point of the maximum accuracy and the cost approach is very subjective depending on the decision maker and the battlefield situation. In addition, the most preferable evaluation method is RPD with a combined array due to its superior performance outside of the design space. In the final analysis, we need a detector/classifier pair that has good performance to minimize error cost and maximize label accuracy.

For further research, we can apply accuracy priorities or mission priorities before we determine the subjective weights of the responses. Though this effort simplifies the CID model by making several simplifying assumptions, we may add a non-declaration choice [17], a cooperative identification process, a decision and firing stage, or, perhaps, a continuous variable associated with time to the current CID model. In addition, by considering the signal and the decision factors that decide the quality of the ROC curve in ROC theory [27], we can approach this problem in a different and inclusive manner.

APPENDIX A: MATLAB® CODE

A. CID Simulation

% This Thesis Code is made by the author.

```
clc;
close all;
clear all;
tic

[TagforReg,Tag]=combarray();

for r=1:size(Tag,1) %for each row in Design matrix  ===================================
   area = zeros(Tag(r,3),1);

   for i=1:Tag(r,4)
      area(i,1)=1;
   end
   area=sortrows(area);
   for i=1:Tag(r,5)
      area(i,1)=2;
   end

   if Tag(r,8)==1
      cost=[1 1 1 1 2]';
   else
      cost=[5 1 1 1 2]';
   end

   d_table = []; %template for detection, 2x2
   c_table = []; %template for classification, 2x3

   d_table = [Tag(r,1), Tag(r,6); 1-Tag(r,1) 1-Tag(r,6)];
   c_table = [Tag(r,2), Tag(r,7) .5; 1-Tag(r,2) 1-Tag(r,7) .5; 0 0 0];


%%%%%% Detection process %%%%%%

   output1 = [];       %for result of detection process
   output2 = [];       %for result of classification process

   column_d = [3 0];   %for detection 0 means nothing, 3 means something
   column_c = [1 2 0]; %for classification 1 means enemy, 2 means friend

   prob = [];
   [I,J] = size(area);
   N = 100;
   cum_confusion = zeros(3);

   for n = 1:N
      confusion = zeros(3);
```

```
for i = 1:I
    for j = 1:J
        if area(i,j) ~= 0
            prob = d_table(:,1);
            [numberchoices,cols] = size(prob);

            out=zeros(2,1);
            out(1,1) = prob(1);

            for k = 2:numberchoices
                out(k,1) = prob(k) + out(k-1,1); %cumulation
            end

            check = 0;
            index = 1;

            while check == 0
                if out(index,1) >= rand(1)                    %comparing threshold with random number
                    output1(i,j) = column_d(index);
                    check = 1;
                else
                    index = index + 1;
                end
            end

        else
            prob = d_table(:,2);
            [numberchoices,cols] = size(prob);

            out=zeros(2,1);
            out(1,1) = prob(1);

            for k = 2:numberchoices
                out(k,1) = prob(k) + out(k-1,1);
            end

            check = 0;
            index = 1;
            while check == 0
                if out(index,1) >= rand(1)
                    output1(i,j) = column_d(index);
                    check = 1;
                else
                    index = index + 1;
                end
            end
        end

        if output1(i,j) == 0

            %confusion matrix for detection
            if area(i,j)==0
                confusion(3,3) = confusion(3,3) + 1;
                output2(i,j) = 0;
```

```
                    end
                    if area(i,j)==1
                        confusion(3,1) = confusion(3,1) + 1;
                        output2(i,j) = 5;
                    end
                    if area(i,j)==2
                        confusion(3,2) = confusion(3,2) + 1;
                        output2(i,j) = 55;
                    end


%%%%%%Classification process %%%%%%

                elseif output1(i,j) ~= 0

                    if area(i,j) == 1

                        prob = c_table(:,1);
                        [numberchoices,cols] = size(prob);
                        gen_prob = rand(1);
                        out(1,1) = prob(1);

                        for k = 2:numberchoices
                            out(k,1) = prob(k) + out(k-1,1);
                        end

                        check = 0;
                        index = 1;
                        while check == 0
                            if out(index,1) >= gen_prob
                                output2(i,j) = column_c(index);
                                check = 1;
                            else
                                index = index + 1;
                            end
                        end

                    elseif area(i,j) == 2

                        prob = c_table(:,2);
                        [numberchoices,cols] = size(prob);
                        out(1,1) = prob(1);

                        for k = 2:numberchoices
                            out(k,1) = prob(k) + out(k-1,1);
                        end

                        check = 0;
                        index = 1;
                        while check == 0
                            if out(index,1) >= rand(1)
                                output2(i,j) = column_c(index);
                                check = 1;
                            else
```

```
                     index = index + 1;
                  end
               end

            elseif area(i,j) == 0

               prob = c_table(:,3);
               [numberchoices,cols] = size(prob);
               out(1,1) = prob(1);

               for k = 2:numberchoices
                  out(k,1) = prob(k) + out(k-1,1);
               end

               check = 0;
               index = 1;
               while check == 0
                  if out(index,1) >= rand(1)
                     output2(i,j) = column_c(index);
                     check = 1;
                  else
                     index = index + 1;
                  end
               end
            end

            %confusion matrix for detection
            if output2(i,j) == 1
               if area(i,j) == 1
                  confusion(1,1) = confusion(1,1) + 1;
               elseif area(i,j) == 2
                  confusion(1,2) = confusion(1,2) + 1;
               elseif area(i,j) == 0
                  confusion(1,3) = confusion(1,3) + 1;
               end
            elseif output2(i,j) == 2
               if area(i,j) == 1
                  confusion(2,1) = confusion(2,1) + 1;
               elseif area(i,j) == 2
                  confusion(2,2) = confusion(2,2) + 1;
               elseif area(i,j) == 0
                  confusion(2,3) = confusion(2,3) + 1;
               end
            end
         end
      end
   end
   cum_confusion = cum_confusion + confusion;
end
CM = cum_confusion / N;

TPR(r,1) = CM(1,1)/sum(CM(:,1)); % vertical analysis P("E"|E)
E1(r,1) = CM(1,2)/sum(CM(1,:)); % horizontal analysis P(F|"E")
E2(r,1) = CM(2,1)/sum(CM(2,:)); % horizontal analysis P(E|"F")
```

```
    E3(r,1) = sum(CM(1:2,3))/sum(sum(CM(1:2,:))); % horizontal analysis P(C|"E" or "F")
    E4(r,1) = sum(CM(3,1:2))/sum(CM(3,:)); % horizontal analysis P(E or F | "C")

    Cost(r,1) = cost(1,1)*CM(2,1)+cost(2,1)*CM(1,2)+cost(3,1)*(CM(1,3)+CM(2,3))
                       +cost(4,1)*(CM(3,1)+CM(3,2))-cost(5,1)*TPR(r,1);

    Accuracy(r,1) = CM(1,1)/sum(sum(CM)) + CM(2,2)/sum(sum(CM)) + CM(3,3)/sum(sum(CM));

end
toc
clear I J i j k r numberchoices howmany prob column_c column_d
```

B. Sub-Function – Generate a Design Matrix

% This Thesis Code is made by the author.

function [TagforReg,Tag] = combarray()

```
clear all
clc;

  load 'new_threshold.mat' threshold_d;
  load 'new_threshold.mat' threshold_c;


  % load 'threshold_mod.mat'; (obtained from RBF)

D = fullfact([100 100 2 2 2 2]);

avector = threshold_d(:,2);
bvector = threshold_c(:,2);
cvector = [100 1000]'; %Map size
dvector = [5 40]'; %number of enemy
evector = [5 40]'; %number of friend
fvector = threshold_d(:,1); %FPR for Detection
gvector = threshold_c(:,1); %FPR for Classification
F = D(:,1);
G = D(:,2);

D = [D,F,G];

for i=1:size(D,1) %sets with test values
    D(i,1)=avector(D(i,1),1);
    D(i,2)=bvector(D(i,2),1);
    D(i,3)=cvector(D(i,3),1);
    D(i,4)=dvector(D(i,4),1);
    D(i,5)=evector(D(i,5),1);
    D(i,7)=fvector(D(i,7),1);
    D(i,8)=gvector(D(i,8),1);
end
```

```
E=D;

for i = 1:size(E,2)
   big = max(E(:,i));
   small = min(E(:,i));
   for j = 1:size(E,1);
       E(j,i) = ((E(j,i)- small)/(big-small))*2 - 1;
   end
end

H = ones(size(E,1),1);
E = [H,E];

Xblock=E(:,1:3);
Xblock=[Xblock,E(:,2).^2, E(:,3).^2, E(:,2).*E(:,3)];

Zblock=E(:,4:7);
XZblock=[E(:,2).*E(:,4),E(:,2).*E(:,5),E(:,2).*E(:,6),E(:,2).*E(:,7),E(:,3).*E(:,4),E(:,3).*E(:,5),E(:,3).*E(:,6
),E(:,3).*E(:,7)];
E = [Xblock, Zblock, XZblock];

X = D;
Y = E;

Tag = [D(:,1:5),D(:,7:8),D(:,6)];
TagforReg = E;

end
```

C. Regression

```
% This Thesis Code is provided by Todd Burnworth, after that...
% Capt Taeho Kim modified for his analysis.

%inputs are A, Response, and Vnames                    <----------user input
%it doesnn't matter if A has leading ones

clc;
clear Bhat Yhat e SSres MSres SSreg MSreg SSt Fo Fstat alpha C H X r d;
clear ePRESS Si2 Rstud t nvector groupnum Ybarvector SSpe ANOVA Xhatp;
clear Yhatp U Z xi xerror yerror Tcrit BoxCoxusedlamda BoxCoxusedlog;
clear leveragepoints Cooks DFFITS Cooksinfluence DFFITSinfluence;
clear DFBETASinfluence DFBETAS DFBETAcountries V R Z Rstud ePRESS;
clear Yhata PRESS;

%%%%%%%%%%%%%%%%%%Switches
   GRAPHS=1;% 0 is off                    <----------user input
   BOXCOX=0;% 0 is off                    <----------user input
   ALLREG=0;% 0 is off                    <----------user input
   LofFit=0;% 0 is off                    <----------user input
   Warnng=0;% 0 is off                    <----------user input
   GENLSQ=0;% 0 is off                              <----------user input

%%%%%%%%add a column of ones to A if it needs one and get sizes of A (n by p)
   Y=Response;
   n=size(A,1);
   if A(:,1)~=ones(n,1)
      A=[ones(n,1) A];
   end
   p=size(A,2);
   globalp=p;
   Filter = int8(ones(1,p));

   %Filter out certain regressors - uncomment to "eliminate"
%     Filter(1,1)=0;% filter B0                      <----------user input*
%     Filter(1,2)=0;% filter B1                      <----------user input
%     Filter(1,3)=0;% filter B2                      <----------user input*
%     Filter(1,4)=0;% filter B3                      <----------user input*
%     Filter(1,5)=0;% filter B4                      <----------user input
%     Filter(1,6)=0;% filter B5                      <----------user input*
%     Filter(1,7)=0;% filter B6                      <----------user input
%     Filter(1,8)=0;% filter B7                      <----------user input

   X=A;
   for i=p:-1:1
      if Filter(1,i)==0
         X(:,i) = [];
      end
   end
   p=size(X,2);

   explist=ones(1,p);
   Xform=int8(zeros(1,p));
   %Pick regressors to transform - uncomment to Xform via Box-Tidwell
%%%%%%%%%%%%%Do not transform x0 via Box Tidwell
%         Xform(1,2)=1;% Xforms x1 via Box-Tidwell            <----------user input
%         Xform(1,3)=1;% Xforms x2 via Box-Tidwell            <----------user input
%         Xform(1,4)=1;% Xforms x3 via Box-Tidwell            <----------user input
%         Xform(1,5)=1;% Xforms x4 via Box-Tidwell            <----------user input
%         Xform(1,6)=1;% Xforms x5 via Box-Tidwell            <----------user input
%         Xform(1,7)=1;% Xforms x6 via Box-Tidwell            <----------user input
%         Xform(1,8)=1;% Xforms x7 via Box-Tidwell            <----------user input

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if Warnng==0
   warning off;
end

%%%%%%%%%%%%%%%%Least Squares
if GENLSQ==1
   Save=X;
   V=cov(X');
   invV=(V)^-1;
   Bhatz=((X'*invV*X)^-1)*X'*invV*Y;
```

```
    K=(V)^.5;%  <--------- if covariances are negative, sqrts will be imaginary.
    Bee=((K)^-1)*X;
    bigZ=Bee*Bhatz; % <-------------also imaginary

    SSresz=bigZ'*bigZ-Bhatz'*Bee'*bigZ;
    MSresz=SSresz/(n-p);

    SSregz=Bhatz'*Bee'*bigZ;
    MSregz=SSregz/(p-1);

    SStz=bigZ'*bigZ;

    %Calculate F statistic for model
    alpha=.90;
    Foz=MSregz/MSresz;
    Fstatz=finv(alpha,p-1,n-p);
    Fpvaluez=1-fcdf(Foz,p-1,n-p);

    %R-squared
    R2z=SSregz/SStz;
    R2adjz=1-(SSresz/(n-p))/(SStz/(n-1));

    %Build table (see pg 80 in book for explanation)
    glmANOVA=zeros(4,6);
    glmANOVA(1,1)=SSregz;  glmANOVA(1,2)=p-1;
    glmANOVA(1,3)=MSregz;
    glmANOVA(1,4)=Foz; glmANOVA(1,5)=Fpvaluez;
    glmANOVA(2,1)=SSresz;  glmANOVA(2,2)=n-p;
    glmANOVA(2,3)=MSresz;
    glmANOVA(3,1)=SStz;    glmANOVA(3,2)=n-1;
    glmANOVA(4,1)=R2z;     glmANOVA(4,2)=R2adjz;


    clear  invV K Bee;
    X=Save;
end
%%%%%%%%%%%%%%%transformations on X -BoxTidwell
    alpha=.9;%                                   <----------user input
    y=Y;

    leading=ones(n,1);
    for i=1:p

        if Xform(1,i)==1
            x=[leading, X(:,i)];
            px=size(x,2);
            a=1;
            olda=10;

            while abs(olda-a)>.00005
                %step 1
                bhat=((x'*x)\eye(px))*x'*y;
                yhat=x*bhat;
                C=(x'*x)\eye(px);
                SSres=y'*y-bhat'*x'*y;
                MSres=SSres/(n-px);
                To=abs(bhat(px,1)/sqrt(MSres*C(px,px)));
                Tcrit=tinv((alpha+(1-alpha)/2),n-px);

                %step 2
                w=x(:,px).*log(x(:,px));
                xw=[x,w];

                %step 3
                bhatw=((xw'*xw)\eye(px+1))*xw'*y;
                yhatw=xw*bhatw;

                %step 4
                Cx=(xw'*xw)\eye(px+1);
                SSresx=y'*y-bhatw'*xw'*y;
                MSresx=SSresx/(n-(px+1));

                Tox=abs(bhatw(px+1,1)/sqrt(MSresx*Cx(px+1,px+1)));
                Tcritx=tinv((alpha+(1-alpha)/2),n-(px+1));

                %step 5
                if To>Tcrit && Tox>Tcritx
                   a=bhatw(px+1,1)/bhat(px,1)+a;
                else
                   olda=a;
                end

                %step 6
                x(:,px)=x(:,px).^a;

            end
            explist(1,i)=a;
        end
    end

    for i=1:p
    explist(1,i)=round(explist(1,i)*2)/2;

        if explist(1,i)>2
            explist(1,i)=2;
        end
        if explist(1,i)<(-2)
            explist(1,i)=(-2);
        end
    end

    for i=1:p
        X(:,i)=X(:,i).^explist(1,i);
    end

clear x y olda To Tcrit Tox Tcritx w Cx bhatw;
clear MSresx SSresx MSres SSres yhatw bhat a xw yhat;
clear Xform leading %explist;
```

```
%%%%%%%%%%%%%%%%transformations on Y -BoxCox
  if BOXCOX==1
    lamda=linspace(-2,2,21);
    lp=size(lamda,2);

    ydot=exp((1/n)*sum(log(Y)));

    for i=1:lp
      if lamda(1,i)~=0
        ytemp=(Y.^lamda(1,i)-1)./(lamda(1,i).*ydot^(lamda(1,i)-1));
      else
        ytemp=ydot.*log(Y);
      end
      bhat=((X'*X)\eye(p))*X'*ytemp;
      yhat=X*bhat;
      C=inv(X'*X);
      SSreslamda(1,i)=ytemp'*ytemp-bhat'*X'*ytemp;
    end

    lmin=min(SSreslamda);
    for i=1:lp
      if SSreslamda(1,i)==lmin
        location=i;
      end
    end
    if lmin~=0
      Y=(Y.^lamda(1,location)-1)/lamda(1,location);
      BoxCoxusedlamda=lamda(1,location)
    else
      Y=log(Y);
      BoxCoxusedlog=1
    end
    if GRAPHS==1
    figure(1)
    scatter(lamda,SSreslamda,'or', 'MarkerFaceColor','c');
    xlabel('Power Transformation Parameter Lamda');
    ylabel('SS_r_e_s'); title('SS_r_e_s vs. Lambda');
    end
  end
clear lp lmin ytemp location bhat yhat SSreslamda lamda ydot;

%%%%%%%%%%%%%%%fit model
  Bhat=((X'*X)\eye(p))*X'*Y;
  Yhat=X*Bhat;

%%%%%%%%%%%%All possible regressions (p counts the intercept)
  if ALLREG==1
    clear All Nines Btemp mm nn U pall Bhata;

    AllReg=zeros(1,p);

    for i=1:p
      cmb=combntns(1:p,i);
```

```
   mm=size(cmb,1);
   nn=size(cmb,2);
   Btemp=zeros(mm,p);
   for j=1:mm
      for k=1:nn
         Btemp(j,cmb(j,k))=1;
      end
   end
   AllReg=[AllReg;Btemp];
end

clear mm nn;
mm=size(AllReg,1);
nn=size(AllReg,2);

U=X; %U holds the original X
for i=1:mm
   for j=nn:-1:1
      if AllReg(i,j)==0
         X(:,j) = [];
      end
   end

   pall=size(X,2);
   Bhata=((X'*X)\eye(pall))*X'*Y;
   Yhata=X*Bhata;
    e=Y-Yhata;
    H=X*((X'*X)\eye(pall))*X';
      for s=1:n
         ePRESS(s,1)=(e(s,1)/(1-H(s,s)))^2;
      end

   All(i,1)=Bhata'*X'*Y -(Y'*ones(n,1))^2/n;             %SSreg
   All(i,2)=Y'*Y-Bhata'*X'*Y;                            %SSres
   All(i,3)=All(i,1)+All(i,2);                          %SSt
   All(i,4)=All(i,1)/All(i,3);                         %R2
   All(i,5)=1-(All(i,2)/(n-pall))/(All(i,3)/(n-1));      %R2adj
   All(i,6)=sum(ePRESS);                                %PRESS

   X=U;
end
X=U;  %reset X

numrgs=sum(AllReg')';
tempM=ones(1,6);
PandR2s=zeros(1,3);

for i=1:p
   k=1;
   for j=1:mm
      if numrgs(j,1)==i
         tempM(k,:)=All(j,:);
         k=k+1;
      end
```

```
          end
            pickbiggest=max(tempM ,[] ,1);
            PandR2s(i,1)=i;               %the # of parameters used
            PandR2s(i,2)=pickbiggest(1,4); %R2
            PandR2s(i,3)=pickbiggest(1,5); %R2adj
        end

        if GRAPHS==1
        figure(2)
        plot(PandR2s(:,1),PandR2s(:,2),'r:o')
        hold on
        plot(PandR2s(:,1),PandR2s(:,3),'b:+')
        hold off
        xlabel('Number of Regression Coeficients');
        ylabel('R^2'); title('R^2 vs. Number of Regression Coefficients');
        legend('R^2','R^2 Adj.',2);
        end

        Nines=ones(mm,1)*9999999;
        All=[AllReg,Nines,All];
    else
        clear All;
    end
    clear nn mm nopt i j k Bhata Nines U pall cmb AllReg Btemp numrgs tempM;
    clear pickbiggest PandR2s;

%%%%%%%%%%%%%%%%%%%%perform ANOVA
    alpha=.95;%                                   <----------user input

    C=(X'*X)\eye(p);

    SSres=Y'*Y-Bhat'*X'*Y;
    MSres=SSres/(n-p);

    SSreg=Bhat'*X'*Y-(Y'*ones(n,1))^2/n;
    MSreg=SSreg/(p-1);

    SSt=SSreg+SSres;

    %Calculate F statistic for model
    Fo=MSreg/MSres;
    Fstat=finv(alpha,p-1,n-p);
    Fpvalue=1-fcdf(Fo,p-1,n-p);

    %Perform marginal T test for each Bhat
    for i=1:p
        To(i,1)=Bhat(i,1)/sqrt(MSres*C(i,i));
        StdErr(i,1)=sqrt(MSres*C(i,i));
        Tcrit(i,1)=tinv((alpha+(1-alpha)/2),n-p);
        Tpvalue(i,1)=2*(1-tcdf(abs(To(i,1)),n-p));
    end

    %R-squared
    R2=SSreg/SSt;
```

```
   R2adj=1-(SSres/(n-p))/(SSt/(n-1));

   %Multicollinearity
%   Z=X;
%   Z(:,1)=[];

%   invR=corr(Z)\eye(p-1);
%    VIF=zeros(p,1);
%    for i=1:p-1
%        VIF(i+1,1)= invR(i,i);
%    end



   for i=1:p
      CIforBhat(i,1)=Bhat(i,1)-tinv((alpha+(1-alpha)/2),n-p)*sqrt(MSres*C(i,i));
      CIforBhat(i,2)=Bhat(i,1);
      CIforBhat(i,3)=Bhat(i,1)+tinv((alpha+(1-alpha)/2),n-p)*sqrt(MSres*C(i,i));
   end

   %Build table (see pg 80 in book for explanation)
   ANOVA=zeros(5+p,6);
   ANOVA(1,1)=SSreg;  ANOVA(1,2)=p-1;    ANOVA(1,3)=MSreg;  ANOVA(1,4)=Fo;
ANOVA(1,5)=Fpvalue;
   ANOVA(2,1)=SSres;  ANOVA(2,2)=n-p;    ANOVA(2,3)=MSres;
   ANOVA(3,1)=SSt;    ANOVA(3,2)=n-1;
   ANOVA(4,1)=R2;     ANOVA(4,2)=R2adj;
   for i=1:p
      ANOVA(5+i,1)=Bhat(i,1);
      ANOVA(5+i,2)=StdErr(i,1);
      ANOVA(5+i,3)=To(i,1);
      ANOVA(5+i,4)=Tcrit(i,1);
      ANOVA(5+i,5)=Tpvalue(i,1);
%       ANOVA(5+i,6)=VIF(i,1);
   end


clear n p Filter Si2 SSres MSres SSreg MSreg SSt Fo Fstat ePRESS i r d t;
clear alpha disp residuals H Fpvalue C R2 R2adj dfssres dfsspe dfsslof;
clear nvector ttlvector Ybarvector m j N groupnum counter lofFo e;
clear lofFpvalue SSlof SSpe StdErr To Tstat Tpvalue Bhat Rstud I VIF;
clear invR Tcrit X LofFit ALLREG BOXCOX GRAPHS globalp Warnng jvector;
clear DFFITS Cooks GENLSQ Foz Fpvaluez SStz SSresz SSregz MSresz MSregz;
clear Yhata Bhata Fstatz R2z R2adjz Save s;
```

D. Confirmation experiment

This Thesis Code is made by the author.

```
clear all;
close all;
```

```
clc;

a1 = input('Input optimal solution # of method 1:   ');
a2 = input('Input optimal solution # of method 2:   ');
a3 = input('Input optimal solution # of method 3:   ');
a4 = input('Input optimal solution # of method 4:   ');
a5 = input('Input optimal solution # of method 5:   ');
a6 = input('Input optimal solution # of method 6:   ');

[TagforReg, Tag, cvector, dvector, evector]=combarray_confirm();

Tag_conf = [Tag(1:10000,1:2),Tag(1:10000,6:7)];
Tag_conf = [Tag_conf(a1,:);Tag_conf(a2,:);Tag_conf(a3,:);Tag_conf(a4,:);Tag_conf(a5,:);
Tag_conf(a6,:)];

c = ceil((cvector(2,1)-cvector(1,1))/4);
d = ceil((dvector(2,1)-dvector(1,1))/4);
e = ceil((evector(2,1)-evector(1,1))/4);

new_cvector = [2*c, c, 3*c, 40*c, 2000]'; %Map size
new_dvector = [2*d, d, 3*d, 40*d, 10]'; %number of enemy
new_evector = [2*e, e, 3*d, 40*d, 70]'; %number of friend
cost_coef = [3 1 1 1 2; 2 1 1 1 2; 4 1 1 1 2; 40 1 1 1 2 ; 10 1 1 1 2]; %cost coefficient

for m = 1:size(new_cvector,1) %ROI for Confirmation experiment

   for r = 1:size(Tag_conf,1) %Threshold needed to confirm

      d_table = [Tag_conf(r,1), Tag_conf(r,3); 1-Tag_conf(r,1) 1-Tag_conf(r,3)];
      c_table = [Tag_conf(r,2), Tag_conf(r,4) .5; 1-Tag_conf(r,2) 1-Tag_conf(r,4) .5; 0 0 0];
      area = zeros(new_cvector(m,1),1); %Map size

      for j=1:new_dvector(m,1) % Enemy
         area(j,1)=1;
      end
      area=sortrows(area);

      for j=1:new_evector(m,1) % Friend
         area(j,1)=2;
      end

      cost = cost_coef(m,:)';
      output1 = [];       %for result of detection process
      output2 = [];       %for result of classification process
      column_d = [3 0];   %for detection 0 means nothing, 3 means something
      column_c = [1 2 0]; %for classification 1 means ET, 2 means FT

      prob = [];
      [I,J] = size(area);
      N = 1000;
      cum_confusion = zeros(3);

      for n = 1:N
         confusion = zeros(3);
```

```
for i = 1:I
    for j = 1:J
        if area(i,j) ~= 0
            prob = d_table(:,1);
            [numberchoices,cols] = size(prob);

            out=zeros(2,1);
            out(1,1) = prob(1);

            for k = 2:numberchoices
                out(k,1) = prob(k) + out(k-1,1); %cumulative distribution function
            end
            check = 0;
            index = 1;

            while check == 0
                if out(index,1) >= rand(1) %search for the probability that is large enough
                    output1(i,j) = column_d(index);
                    check = 1;
                else
                    index = index + 1;
                end
            end

        else
            prob = d_table(:,2);
            [numberchoices,cols] = size(prob);

            out=zeros(2,1);
            out(1,1) = prob(1);

            for k = 2:numberchoices
                out(k,1) = prob(k) + out(k-1,1);
            end

            check = 0;
            index = 1;
            while check == 0
                if out(index,1) >= rand(1)
                    output1(i,j) = column_d(index);
                    check = 1;
                else
                    index = index + 1;
                end
            end
        end

        if output1(i,j) == 0

            %confusion matrix for detection
            if area(i,j)==0
                confusion(3,3) = confusion(3,3) + 1;
                output2(i,j) = 0;
            end
        if area(i,j)==1
            confusion(3,1) = confusion(3,1) + 1;
            output2(i,j) = 5;
        end
        if area(i,j)==2
            confusion(3,2) = confusion(3,2) + 1;
            output2(i,j) = 55;
        end

    elseif output1(i,j) ~= 0
        if area(i,j) == 1
            prob = c_table(:,1);
            [numberchoices,cols] = size(prob);
            gen_prob = rand(1);
            out(1,1) = prob(1);

            for k = 2:numberchoices
                out(k,1) = prob(k) + out(k-1,1);
            end

            check = 0;
            index = 1;
            while check == 0
                if out(index,1) >= gen_prob
                    output2(i,j) = column_c(index);
                    check = 1;
                else
                    index = index + 1;
                end
            end

        elseif area(i,j) == 2

            prob = c_table(:,2);
            [numberchoices,cols] = size(prob);
            out(1,1) = prob(1);

            for k = 2:numberchoices
                out(k,1) = prob(k) + out(k-1,1);
            end

            check = 0;
            index = 1;
            while check == 0
                if out(index,1) >= rand(1)
                    output2(i,j) = column_c(index);
                    check = 1;
                else
                    index = index + 1;
                end
            end

        elseif area(i,j) == 0
```

```
                prob = c_table(:,3);
                [numberchoices,cols] = size(prob);
                out(1,1) = prob(1);

                for k = 2:numberchoices
                   out(k,1) = prob(k) + out(k-1,1);
                end

                check = 0;
                index = 1;
                while check == 0
                   if out(index,1) >= rand(1)
                      output2(i,j) = column_c(index);
                      check = 1;
                   else
                      index = index + 1;
                   end
                end
             end

             %confusion matrix for classification
             if output2(i,j) == 1
                if area(i,j) == 1
                   confusion(1,1) = confusion(1,1) + 1;
                elseif area(i,j) == 2
                   confusion(1,2) = confusion(1,2) + 1;
                elseif area(i,j) == 0
                   confusion(1,3) = confusion(1,3) + 1;
                end
             elseif output2(i,j) == 2
                if area(i,j) == 1
                   confusion(2,1) = confusion(2,1) + 1;
                elseif area(i,j) == 2
                   confusion(2,2) = confusion(2,2) + 1;
                elseif area(i,j) == 0
                   confusion(2,3) = confusion(2,3) + 1;
                end
             end
          end
       end
    end
    cum_confusion = cum_confusion + confusion;
end
CM = cum_confusion / N;

TPR(r,m) = CM(1,1)/sum(CM(:,1)); % vertical analysis P("E"|E)
E1(r,m) = CM(1,2)/sum(CM(1,:)); % horizontal analysis P(F|"E")
E2(r,m) = CM(2,1)/sum(CM(2,:)); % horizontal analysis P(E|"F")
E3(r,m) = sum(CM(1:2,3))/sum(sum(CM(1:2,:))); % horizontal analysis P(C|"E" or "F")
E4(r,m) = sum(CM(3,1:2))/sum(CM(3,:)); % horizontal analysis P(E or F | "C")

Cost(r,m) = cost(1,1)*CM(2,1)+cost(2,1)*CM(1,2)+cost(3,1)*(CM(1,3)+CM(2,3))
               + cost(4,1)*(CM(3,1)+CM(3,2))-cost(5,1)*TPR_r(r,1);
Accuracy(r,m) = CM(1,1)/sum(sum(CM)) + CM(2,2)/sum(sum(CM)) + CM(3,3)/sum(sum(CM));
```

```
    end
end
```

E. Radial Basis Functions – Create ROC curves

```
% This Thesis Code is provided by Todd Paciencia [16], after that...
% Capt Taeho Kim modified for his analysis.
```

<u>function[newpts,data]=createsurrogate(regtype,X1,X2,kerneltype,polytype,numnewpts)</u>

```
%polytypes are regpoly0, regpoly1, regpoly2, regpoly2reduced, regpoly3,
%regpoly3reduced

dvmatrix=[];
dvmatrix(1)=min(X1); %General surrogate
dvmatrix(2)=max(X2);

if strcmp(regtype,'rbf')==1
   %kernel types: bi-harmonic, tri-harmonic, multiquadric, invmultiquadric, thinplatespline, gaussian
   rbfmodel=buildRBF(X1,X2,kerneltype,polytype);
   newpts=genpts(numnewpts,1,X1);
       for i=1:size(newpts,1)
       fx(i)=evalRBF(newpts(i,:)',rbfmodel);  %input is col
       end
       data=fx;

elseif strcmp(regtype,'nw')==1
     %Just set hmin,hmax
   hmin=0.1; %A lot of curvature to get to points
   hmax=50; %Assuming not all responses are same, was 3
   h=mean(hmin,hmax);
   %kernel types: gaussian, uniform,triangle,epanechnikov,quartic,triweight,cosinus
   nwmodel=buildNW(X1,X2,kerneltype,h,hmin,hmax);
%nwmodel.sigma
    newpts=genpts(numnewpts,1,X1);
      for i=1:size(newpts,1)
       fx(i)=evalNW(newpts(i,:)',nwmodel);
          if fx(i)==1/eps
             fx(i)=NaN;
          end
       end
      data=fx;

elseif strcmp(regtype,'dace')==1
   %kerneltype is really corr type
   [s1,s2]=size(X1);
   thetaint=10*ones(1,s2);
   [upb,lob,initialtheta] = thetabds(X1,X2,polytype,kerneltype,thetaint,0);
   [dmodel,perf]=dacefit(X1,X2,polytype,kerneltype,initialtheta,lob,upb);
```

```
  %dmodel.theta %theta vector
   %Corr fn types (kerneltype): corrgauss, corrcubic, correxp, correxpg, corrlin,
   %corrspherical, corrspline
  newpts=genpts(numnewpts,1,X1);
      for i=1:size(newpts,1)
      fx(i)=predictor(newpts(i,:),dmodel);
      end
      data=fx;
end
```

function rbfmodel = buildRBF(S,Y,typeKernel,poly)

```
%BUILDRBF  Build a surrogate function based on Radial Basis Functions

%******************************************************************************
% buildRBF: Builds a surrogate based on radial basis functions.
% ----------------------------------------------------------------------------
% VARIABLES:
%   S           = matrix of data sites, stored row-wise
%   Y           = column vector of responses, each corresponding to a data site
%   typeKernel  = string indicating the type of kernel used in the RBF
%   rbfmodel    = structure of parameters that define the RBF estimator
%     .kernel   =   type of kernel used in the RBF estimator
%     .S        =   matrix of data sites
%     .coeff    =   vector of polynomial coefficients
%   nSites      = number of data sites
%   n           = number of variables
%   r           = matrix of distances between data sites
%   A           = system matrix
%******************************************************************************
[nSites, n] = size(S);
r = zeros(nSites,nSites);
for i = 1:nSites
   for j = 1:i-1
      r(i,j) = norm(S(i,:) - S(j,:));
   end
   r(1:i-1,i) = r(i,1:i-1);
end
%Added
if strcmp(typeKernel,'multiquadric') || strcmp(typeKernel,'invmultiquadratic') ||
strcmp(typeKernel,'gaussian') ==1
   c = mean(mean(r)); %Avg dist between centers
   rbfmodel.c = c;
else
   %Not used, could be anything
   c=1;
   rbfmodel.c = 1;
end
%Note, you must use either x2fx or regpoly in both build and eval, o.w.
%coeffs change order
%Also, regpoly1 is present, skips the if block and leaves S alone
check=0;
```

```
if strcmp(poly,'regpoly2')==1
   S_orig = S;
   S = x2fx(S,'quadratic');
   S(:,1)=[]; %get rid of constant term, is added later
   n=size(S,2);
   check=1;
elseif strcmp(poly,'regpoly2reduced')==1
   S_orig = S;
   S = x2fx(S,'purequadratic');
   S(:,1)=[]; %get rid of constant term, is added later
   n=size(S,2);
   check=1;
elseif strcmp(poly,'regpoly3')==1
   S_orig = S;
   S = regpoly3(S);
   S(:,1)=[]; %get rid of constant term, is added later
   n=size(S,2);
   check=1;
elseif strcmp(poly,'regpoly3reduced')==1
   S_orig = S;
   S = regpoly3reduced(S);
   S(:,1)=[]; %get rid of constant term, is added later
   n=size(S,2);
   check=1;
elseif strcmp(poly,'regpoly0')==1
   S_orig = S;
   S=[];
   n=0;
   check=1;
end

%----
A = [kernelRBF(typeKernel, r, c), ones(nSites,1), S;
   [ones(nSites,1), S]',      zeros(n+1,n+1)];
%**************************************************************************
%REPAIRING ILL-CONDITIONED MATRIX USING SINGULAR VALUE DECOMPOSITION IF
%REPAIR IS NEEDED, ELSE COEFFICIENTS ARE COMPUTED VIA LU FACTORIZATION
%**************************************************************************
if condest(A)>=1/eps%1e15
   disp('Ill-conditioned, repairing')
   [U,S2,V] = svd(A);
   s = diag(S2);
   e = zeros(length(s),1);
   ind = s/max(abs(s)) >= eps;%1e-8;
   e(ind) = 1./s(ind);

   E = U*diag(e)*V';
   rbfmodel.coeff  = E * [Y(:); zeros(n+1,1)];
else
   rbfmodel.coeff  = A \ [Y(:); zeros(n+1,1)];
end
%**************************************************************************
%END SVD IF IT WAS NEEDED, ELSE COEFFICIENTS WERE COMPUTED VIA LU
%FACTORIZATION
```

```
%*************************************************************************
rbfmodel.kernel = typeKernel;

if check==1
   S=S_orig;
end
rbfmodel.poly = poly;
%------
rbfmodel.S      = S;


return
```

function fx = evalRBF(x,rbfmodel)

```
%EVALRBF  Evaluate a radial basis function surrogate function at a given point

%*****************************************************************************
% evalRBF: Evaluates a radial basis function surrogate at a given point.
% -----------------------------------------------------------------------------
% Calls:     kernel
% VARIABLES:
%   fx          = RBF value at x
%   x           = the point to be evaluated
%   rbfmodel    = structure of all parameters that define the RBF surrogate
%     .S        =   matrix of data sites
%     .kernel   =   string indicating the choice of kernel function
%     .coeff    =   coefficients that define the RBF
%   nSites      = number of data sites
%   n           = number of variables
%   r           = vector of distances from x to data sites
%*****************************************************************************
[nSites,n] = size(rbfmodel.S);
r = zeros(nSites,1);
for j = 1:nSites
   r(j) = norm(x - rbfmodel.S(j,:)');
end
%Added
if strcmp(rbfmodel.poly,'regpoly2')==1
   x=x2fx(x','quadratic');
   x=x';
   x(1,:)=[];%Get rid of constant, taken care of later
elseif strcmp(rbfmodel.poly,'regpoly2reduced')==1
   x=x2fx(x','purequadratic');
   x=x';
   x(1,:)=[];%Get rid of constant, taken care of later
elseif strcmp(rbfmodel.poly,'regpoly3')==1
   x=regpoly3(x');
   x=x';
   x(1,:)=[];%Get rid of constant, taken care of later
elseif strcmp(rbfmodel.poly,'regpoly3reduced')==1
   x=regpoly3reduced(x');
   x=x';
   x(1,:)=[];%Get rid of constant, taken care of later
elseif strcmp(rbfmodel.poly,'regpoly0')==1
   x=[]; %Just a constant added to RBFs
end
%--------
y  = [kernelRBF(rbfmodel.kernel,r, rbfmodel.c); 1; x(:)];
fx = rbfmodel.coeff'*y;
if ~isfinite(fx)
   fx = 1/eps;
end
return
```

function [chromosomes] = genpts(pop,numdv,X)

```
%Each chromosome has set of dv values

mindv=min(X);
maxdv=max(X);

%Initialize chromosomes using range in dv space
for i=1:pop

   for j=1:numdv
      chromosomes(i,j)=mindv(j)+(maxdv(j)-mindv(j))*rand(1);
   end

end
```

function K = kernelRBF(typeKernel,x, c)

```
%KERNEL  Evaluate the kernel of an RBF estimator at a given point.

% ---------------------------------------------------------------------------
%  Called by: evalRBF
%  VARIABLES:
%    typeKernal = type of kernel used in the Nadaraya-Watson estimator
%    x          = point to be evaluated (vectors allowed, scalars preferred)
%    K          = kernel value at the point x
%******************************************************************************
switch lower(typeKernel)
   case 'bi-harmonic'
      K = x;
      %Added
   case 'tri-harmonic'
      for i=1:size(x,1) %row
         for j=1:size(x,2) %col
            K(i,j) = x(i,j)^3;
         end
      end
```

```
    case 'multiquadric'
        for i=1:size(x,1) %row
            for j=1:size(x,2) %col
                K(i,j) = (x(i,j)^2+c^2)^.5;
            end
        end
    case 'invmultiquadric'
        for i=1:size(x,1) %row
            for j=1:size(x,2) %col
                K(i,j)=(x(i,j)^2+c^2)^(-.5);
            end
        end
    case 'thinplatespline'
        for i=1:size(x,1) %row
            for j=1:size(x,2) %col
                if x(i,j)==0
                    K(i,j)=0; %otherwise have log of 0
                else
                    K(i,j) = x(i,j)^2*log(x(i,j));
                end
            end
        end
    case 'gaussian'
        for i=1:size(x,1) %row
            for j=1:size(x,2) %col
                K(i,j)=exp(-c*x(i,j)^2);
            end
        end
    otherwise
        error('Invalid kernel type used in Radial Basis Function estimator');
end
return
```

function [f, df] = regpoly1(S)

```
%REGPOLY1  First order polynomial regression function

% Call:    f = regpoly1(S)
%          [f, df] = regpoly1(S)
%
% S : m*n matrix with design sites
% f = [1  s]
% df : Jacobian at the first point (first row in S)

% hbn@imm.dtu.dk
% Last update April 12, 2002

[m n] = size(S);
f = [ones(m,1)  S];
if  nargout > 1
  df = [zeros(n,1) eye(n)];
end
```

function [f, df] = regpoly2(S)

```
%REGPOLY2  Second order polynomial regression function

% Call:    f = regpoly2(S)
%          [f, df] = regpoly2(S)
%
% S : m*n matrix with design sites
% f =  [1 S S(:,1)*S S(:,2)S(:,2:n) ... S(:,n)^2]
% df : Jacobian at the first point (first row in S)

% hbn@imm.dtu.dk
% Last update September 4, 2002

[m n] = size(S);
nn = (n+1)*(n+2)/2;  % Number of columns in f
% Compute  f
f = [ones(m,1) S zeros(m,nn-n-1)];
j = n+1;   q = n;
for  k = 1 : n
  f(:,j+(1:q)) = repmat(S(:,k),1,q) .* S(:,k:n);
  j = j+q;   q = q-1;
end

if  nargout > 1
  df = [zeros(n,1)  eye(n)  zeros(n,nn-n-1)];
  j = n+1;   q = n;
  for  k = 1 : n
    df(k,j+(1:q)) = [2*S(1,k) S(1,k+1:n)];
    for i = 1 : n-k,  df(k+i,j+1+i) = S(1,k); end
    j = j+q;   q = q-1;
  end
end
```

APPENDIX B: ROC THRESHOLD DATA FILE

SET1				SET2			
DETECTOR		CLASSFIER		DETECTOR		CLASSFIER	
FPR__D	TPR_D	FPR_C	TPR_C	FPR__D	TPR_D	FPR_C	TPR_C
0.01	0.1685	0.01	0.0803	0.0005	0.4422	0.0005	0.4082
0.02	0.3483	0.02	0.166	0.001	0.4932	0.001	0.4592
0.03	0.524	0.03	0.2542	0.0015	0.5694	0.0015	0.5354
0.04	0.6816	0.04	0.3431	0.002	0.6098	0.002	0.5758
0.05	0.8	0.05	0.4311	0.0025	0.644	0.0025	0.61
0.06	0.8443	0.06	0.5168	0.003	0.674	0.003	0.64
0.07	0.8656	0.07	0.5987	0.0035	0.684	0.0035	0.65
0.08	0.8825	0.08	0.6751	0.004	0.704	0.004	0.67
0.09	0.8996	0.09	0.7435	0.0045	0.714	0.0045	0.68
0.1	0.917	0.1	0.8	0.005	0.724	0.005	0.69
0.11	0.9268	0.11	0.8375	0.0055	0.754	0.0055	0.72
0.12	0.9321	0.12	0.8629	0.006	0.754	0.006	0.72
0.13	0.9362	0.13	0.8802	0.0065	0.764	0.0065	0.73
0.14	0.9403	0.14	0.8919	0.007	0.7667	0.007	0.7327
0.15	0.944	0.15	0.9	0.0075	0.7865	0.0075	0.7525
0.16	0.948	0.16	0.9067	0.008	0.8261	0.008	0.7921
0.17	0.9517	0.17	0.9118	0.0085	0.8261	0.0085	0.7921
0.18	0.9549	0.18	0.9153	0.009	0.8261	0.009	0.7921
0.19	0.9576	0.19	0.9178	0.0095	0.8379	0.0095	0.8039
0.2	0.96	0.2	0.92	0.01	0.854	0.01	0.82
0.21	0.9627	0.21	0.9231	0.0105	0.8673	0.0105	0.8333
0.22	0.9653	0.22	0.9263	0.011	0.8673	0.011	0.8333
0.23	0.9676	0.23	0.9292	0.0115	0.8771	0.0115	0.8431
0.24	0.9695	0.24	0.9317	0.012	0.8981	0.012	0.8641
0.25	0.971	0.25	0.9339	0.0125	0.8981	0.0125	0.8641
0.26	0.9722	0.26	0.9356	0.013	0.8994	0.013	0.8654
0.27	0.9731	0.27	0.937	0.0135	0.909	0.0135	0.875
0.28	0.9738	0.28	0.9381	0.014	0.9102	0.014	0.8762
0.29	0.9743	0.29	0.9391	0.0145	0.9292	0.0145	0.8952
0.3	0.975	0.3	0.94	0.015	0.9307	0.015	0.8967
0.31	0.9761	0.31	0.9412	0.0155	0.9308	0.0155	0.8972

0.32	0.9773	0.32	0.9425	0.016	0.9308	0.016	0.8972
0.33	0.9786	0.33	0.9437	0.0165	0.9401	0.0165	0.9065
0.34	0.9798	0.34	0.9448	0.017	0.9495	0.017	0.9159
0.35	0.9809	0.35	0.9459	0.0175	0.9595	0.0175	0.9259
0.36	0.9819	0.36	0.9468	0.018	0.9595	0.018	0.9259
0.37	0.9828	0.37	0.9476	0.0185	0.9595	0.0185	0.9259
0.38	0.9836	0.38	0.9484	0.019	0.9595	0.019	0.9259
0.39	0.9843	0.39	0.9492	0.0195	0.9595	0.0195	0.9259
0.4	0.985	0.4	0.95	0.02	0.9595	0.02	0.9259
0.41	0.9857	0.41	0.951	0.0205	0.9595	0.0205	0.9259
0.42	0.9865	0.42	0.952	0.021	0.9595	0.021	0.9259
0.43	0.9871	0.43	0.9531	0.0215	0.9595	0.0215	0.9259
0.44	0.9877	0.44	0.9542	0.022	0.9595	0.022	0.9259
0.45	0.9882	0.45	0.9552	0.0225	0.9694	0.0225	0.9358
0.46	0.9887	0.46	0.9562	0.023	0.97	0.023	0.9364
0.47	0.9891	0.47	0.9572	0.0235	0.9795	0.0235	0.9459
0.48	0.9894	0.48	0.9581	0.024	0.9795	0.024	0.9459
0.49	0.9897	0.49	0.959	0.0245	0.9795	0.0245	0.9459
0.5	0.99	0.5	0.96	0.025	0.9795	0.025	0.9459
0.51	0.9904	0.51	0.961	0.0255	0.9795	0.0255	0.9459
0.52	0.9908	0.52	0.9621	0.026	0.9795	0.026	0.9459
0.53	0.9911	0.53	0.9631	0.0265	0.989	0.0265	0.9554
0.54	0.9915	0.54	0.9641	0.027	0.9891	0.027	0.9558
0.55	0.9918	0.55	0.9651	0.0275	0.9891	0.0275	0.9558
0.56	0.9921	0.56	0.9661	0.028	0.9894	0.028	0.9561
0.57	0.9923	0.57	0.9671	0.0285	0.9894	0.0285	0.9561
0.58	0.9926	0.58	0.9681	0.029	0.9894	0.029	0.9561
0.59	0.9928	0.59	0.969	0.0295	0.9894	0.0295	0.9561
0.6	0.993	0.6	0.97	0.03	0.9894	0.03	0.9561
0.61	0.9932	0.61	0.971	0.0305	0.9894	0.0305	0.9561
0.62	0.9935	0.62	0.972	0.031	0.9894	0.031	0.9561
0.63	0.9937	0.63	0.973	0.0315	0.9894	0.0315	0.9561
0.64	0.994	0.64	0.974	0.032	0.9894	0.032	0.9561
0.65	0.9942	0.65	0.975	0.0325	0.9894	0.0325	0.9561
0.66	0.9944	0.66	0.976	0.033	0.9898	0.033	0.9565
0.67	0.9945	0.67	0.977	0.0335	0.9898	0.0335	0.9565
0.68	0.9947	0.68	0.978	0.034	0.9898	0.034	0.9565
0.69	0.9948	0.69	0.979	0.0345	0.9898	0.0345	0.9565

0.7	0.995	0.7	0.98	0.035	0.9898	0.035	0.9565
0.71	0.9952	0.71	0.9811	0.0355	0.9902	0.0355	0.9569
0.72	0.9953	0.72	0.9821	0.036	0.9902	0.036	0.9569
0.73	0.9955	0.73	0.9832	0.0365	0.9902	0.0365	0.9569
0.74	0.9956	0.74	0.9842	0.037	0.9902	0.037	0.9569
0.75	0.9958	0.75	0.9853	0.0375	0.9902	0.0375	0.9569
0.76	0.9959	0.76	0.9863	0.038	0.9906	0.038	0.9573
0.77	0.9961	0.77	0.9873	0.0385	0.9906	0.0385	0.9573
0.78	0.9962	0.78	0.9883	0.039	0.9906	0.039	0.9573
0.79	0.9964	0.79	0.9892	0.0395	0.9994	0.0395	0.9661
0.8	0.9965	0.8	0.99	0.04	0.9994	0.04	0.9661
0.81	0.9966	0.81	0.9907	0.0405	0.9997	0.0405	0.9664
0.82	0.9967	0.82	0.9914	0.041	0.9997	0.041	0.9664
0.83	0.9969	0.83	0.992	0.0415	0.9997	0.0415	0.9664
0.84	0.997	0.84	0.9926	0.042	0.9997	0.042	0.9664
0.85	0.9971	0.85	0.9931	0.0425	0.9997	0.0425	0.9664
0.86	0.9972	0.86	0.9936	0.043	0.9997	0.043	0.9664
0.87	0.9972	0.87	0.9941	0.0435	0.9997	0.0435	0.9664
0.88	0.9973	0.88	0.9945	0.044	0.9997	0.044	0.9664
0.89	0.9972	0.89	0.9948	0.0445	0.9997	0.0445	0.9664
0.9	0.997	0.9	0.995	0.045	0.9997	0.045	0.9664
0.91	0.9965	0.91	0.9949	0.0455	0.9997	0.0455	0.9664
0.92	0.9958	0.92	0.9948	0.046	1	0.046	0.9667
0.93	0.9952	0.93	0.9946	0.0465	1	0.0465	0.9667
0.94	0.9948	0.94	0.9946	0.047	1	0.047	0.9667
0.95	0.9945	0.95	0.9947	0.0475	1	0.0475	0.9667
0.96	0.9946	0.96	0.995	0.048	1	0.048	0.9667
0.97	0.995	0.97	0.9956	0.0485	1	0.0485	0.9667
0.98	0.996	0.98	0.9965	0.049	1	0.049	0.9667
0.99	0.9976	0.99	0.998	0.0495	1	0.0495	0.9667
1	1	1	1	0.05	1	0.05	0.9667

Bibliography

1. 10 August 2006 Interview by Zito, S, McNickle, C., Craig, F. “Full Transcript of Donald Rumsfeld Interview,” *The post chronicle*, 31 August 2006.

2. Defense Science Board. *Combat Identification.* Report. Washington: Office of the Secretary of Defense for Acquisition and Technology, 1996.

3. Shannon, R.E. *Systems Simulation: The Art and Science*. Englewood Cliffs, N.J.: Prentice-Hall, 1975.

4. Hartman, James K. Course notes, OPER 671, Combat Modeling I, Lecture Notes in Aggregated Combat Modeling. Air Force Institute of Technology, Wright-Patterson AFB OH, 1985.

5. Defense science and technology strategy and plans, “Combat Identification,” 2000. [http://www.wslfweb.org/docs/dstp2000/jwstppdf/08-CID.pdf; Accessed Jul 9, 2007].

6. George Mason University, “Combat Identification with Bayesian Networks.” [http://ite.gmu.edu/~klaskey/papers/LaskeyCCRTSCombatID.pdf; Accessed Aug 4, 2007].

7. Sadowski, C. "Combat Identification." Briefing to students, Air Force Institute of Technology, Wright-Patterson AFB OH. 3 May 2007.

8. Rodriguez, June., PhD Student. “Combat Identification Characterization.” Air Force Institute of Technology, Wright-Patterson AFB OH, 2006.

9. Donohue, Tom and Hylton, Paul. “A/G Requirement.” Briefing to students, Air Force Institute of Technology, Wright-Patterson AFB OH. Mar 1999.

10. Hall, Debbie. “Air Force CID Issues and Analyses.” Briefing to students, Air Force Institute of Technology, Wright-Patterson AFB OH. May 2001.

11. Rice, Roy E. “Quantifying “Persistence” in the Context of Find-Fix-Finish (FFF),” *The Bulletin of Military Operations Research, PHALANX*, 40-2: 11-16 (June 2007)

12. Sanchez, Paul J. “As Simple As Possible, But No Simpler: A Gentle Introduction to Simulation Modeling,” *Winter Simulation Conference* (2006).

13. Macal, Charles M. and North, Michael J. “Tutorial on Agent-Based Modeling and Simulation Part2: How to Model with Agents,” *Winter Simulation Conference* (2006).

14. Answers.com: Online dictionary. [http://www.answers.com/object; Accessed Oct 2007].

15. Fawcett, Tom. "An introduction to ROC analysis," *Pattern Recognition Letters,* 27 (December 2005).

16. Paciencia, Todd J. *Multi-Objective Optimization of Mixed Variable, Stochastic Systems Using Single-Objective Formulations*. MS thesis, AFIT/GOR/ENS/08-17. Department of Operations Research, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, 2008.

17. Laine, Trevor I and Bauer, Jr. Kenneth W. "A Mathematical Framework to Optimize ATR Systems with Non-Declarations and Sensor Fusion," *Computer & Operations Research,* 2006.

18. Alsing, Stephen G. *The evaluation of competing classifiers*. MS thesis, AFIT/DS/ENS/00-01. Department of Operations Research, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, 2000 (ADA375294).

19. Koopman, B.O. *OEG Report 56 on Search and Screening*. Operations Evaluation Group, office of the Chief of Naval Operations, Navy Department, 1946.

20. Miller, John O. Class Slides, OPER 671, Combat Modeling I. Department of Operations Research, Air Force Institute of Technology, Wright-Patterson AFB OH, July 2007.

21. Nocedal Jorge and Wright, Stephen J. *Numerical Optimization*. New York: Springer, 2006.

22. Montgomery, Douglas C. *Design and Analysis of Experiments* (6th Edition). New Jersey: Wiley, 2005.

23. Fowlkes, W. Y. and Creveling, C. M. *Engineering Methods for Robust Product Design: Using Taguchi Methods in Technology and Product Development*. New Jersey: Pearson Education, 1995.

24. Montgomery, Douglas C. *Design and Analysis of Experiments* (3rd Edition). New Jersey: Wiley, 1991.

25. Law Averill M. *Simulation modeling and analysis* (4th Edition). New York: McGraw-Hill Book Company, 2007.

26. Wackerly, Denneis D. and others. *Mathematical statistics with applications* (6th Edition). Pacific Grove CA: Duxbury, 2002.

27. Bauer, Kenneth W. Course Notes, OPER 685, Applied Multivariate Analysis I. Department of Operational Sciences, Air Force Institute of Technology, Wright-Patterson AFB OH. November 2007: 88, 95

Vita

Captain TaeHo Kim graduated from Gwang-Ju high school in Gwang-Ju, Korea. He entered undergraduate studies at the Korea Military Academy where be graduated with a Bachelor of Engineering Degree in Civil Engineering and received a regular commission in March 2003.

He served as a platoon leader, G-1 officer at Engineer Battalion unit and so on for three years. In August 2006 he entered the Graduate School of Operations Research, Air Force Institute of Technology. Upon graduation, he will be assigned to be a company commander.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 074-0188*

1. REPORT DATE *(DD-MM-YYYY)* 07-03-2008	**2. REPORT TYPE** Master's Thesis	**3. DATES COVERED** *(From – To)* Jun 2007 - Mar 2008

4. TITLE AND SUBTITLE COMBAT IDENTIFICATION MODELING USING ROBUST OPTIMIZATION TECHNIQUES	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Kim, TaeHo, Captain, ROKA	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S) Air Force Institute of Technology, Graduate School of Engineering and Management (AFIT/EN), 2950 Hobson Street, Building 642, WPAFB OH 45433-7765	**8. PERFORMING ORGANIZATION REPORT NUMBER** AFIT/GOR/ENS/08-11
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) ACC/A8SI, AFOSR, Attn: Charles Sadowski Jr., 204 Dodd Blvd Suite 226, Langley AFB, Va 23665-2702	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The purposes of this research were: (1) the modeling of a CID situation and (2) the search for robust and controllable input variable settings. The inputs were defined as controllable and noise variables and the confusion matrices in ROC theory were adapted to act as controllable factors. In this research a simple virtual battlespace representation is employed. The experimental results of the CID system are summarized by a posterior confusion matrix and throughout the confusion matrix analysis we can obtain all various types of data such as accuracy, error cost, error rates, and so forth. To find the optimal parameters three evaluation techniques were applied: (1) Linearly constrained discrete optimization, (2) Taguchi's S|N ratio method and (3) Robust parameter design with a combined array. The results are compared and contrasted across different objective functions.

15. SUBJECT TERMS

Combat Modeling, Combat Identification, Receiver Operating Characteristics Analysis, Confusion Matrix, Robust Parameter Design, Linearly Constrained Optimization, Taguchi's Signal to Noise Ratio, Accuracy

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Dr. Kenneth W. Bauer
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			**19b. TELEPHONE NUMBER** *(Include area code)*
U	U	U	UU	125	(937) 255-6565, ext 4328; e-mail: Kenneth.Bauer@afit.af.mil

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18